AN5365

SAM L11 安全参考指南

简介

本文档旨在帮助开发人员使用 SAM L11 安全功能编译安全嵌入式应用程序。

本文档涵盖应用程序开发的以下几个方面：

- 单开发人员和双开发人员方法
- 使用 SAM L11 生态系统开发安全解决方案
- 使用 ARMv8-M 的 Arm® TrustZone®技术以及调试访问级别保护安全软件
- 使用安全引导确保系统可信根

下列内容通过裸机软件示例说明了如何使用密钥安全功能：

- 使用 SAM L11 安全、非安全和混合安全外设
- 针对 AES-128、SHA-256 和 GCM 算法使用嵌入式加密加速器（CRYA）
- 使用数据闪存和可信 RAM 存储应用程序机密信息，并通过篡改检测、加扰和静默访问功能加以保护

目录

1. SAM L11 安全功能简介

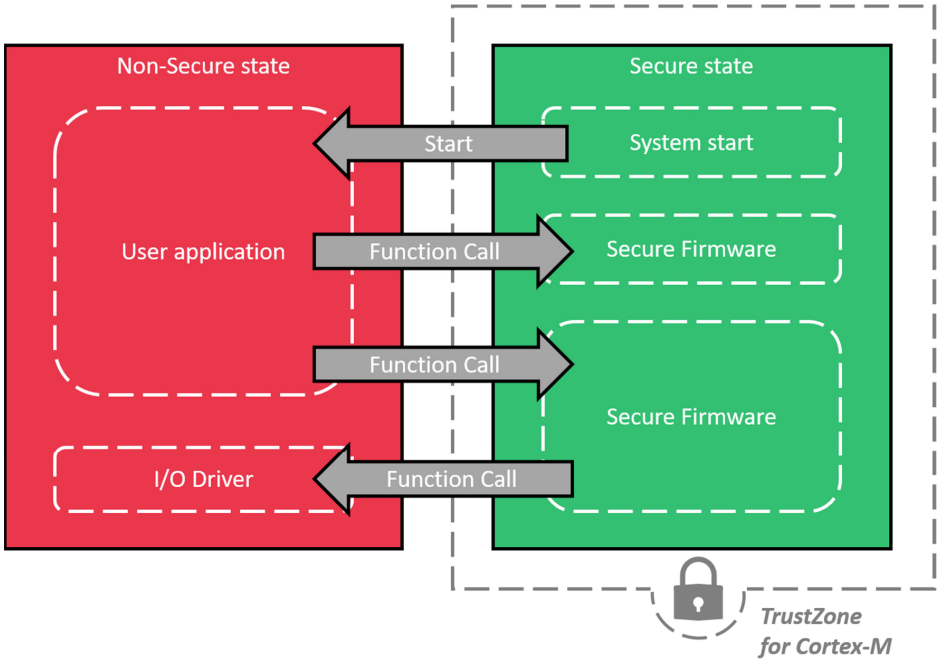
1.1 ARMv8-M 的 TrustZone 技术

Microchip SAM L11 单片机（MCU）的中央安全元件是 ARMv8-M 器件实现的 TrustZone 技术。TrustZone 技术是一种片上系统（System-on-Chip，SoC）和 MCU 系统范围的安全方法，支持在单个 MCU 上运行安全和非安全应用程序代码。

ARMv8-M 器件的 TrustZone 技术基于 Cortex®-M23 内核中实现的特定硬件，该硬件与专用安全指令集结合使用。它支持创建多个软件安全域，限制只有可信软件才能访问选定存储器、外设和 I/O，而不会影响系统性能。

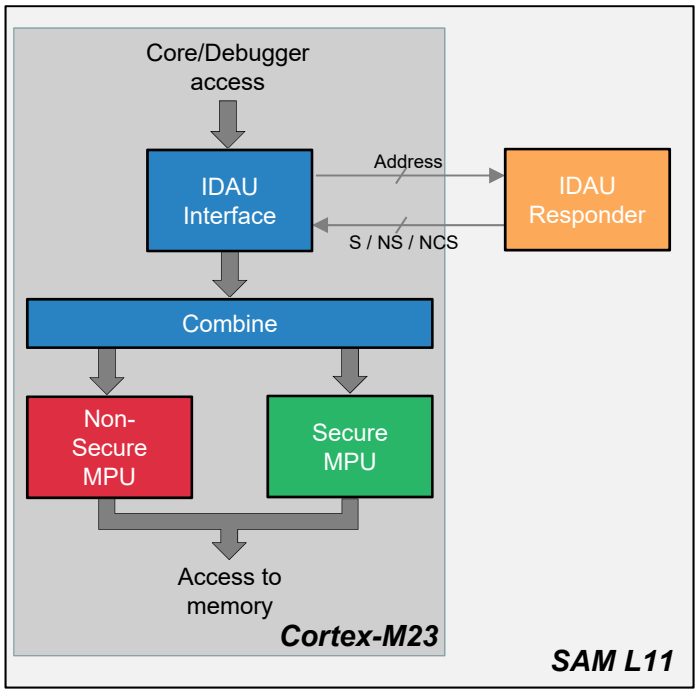
ARMv8-M 器件的 TrustZone 技术主要旨在简化深度嵌入式器件的安全评估。ARMv8-M 的 TrustZone 嵌入式软件应用程序的原理如下图所示。

图 1-1. 安全状态与非安全状态之间的标准交互



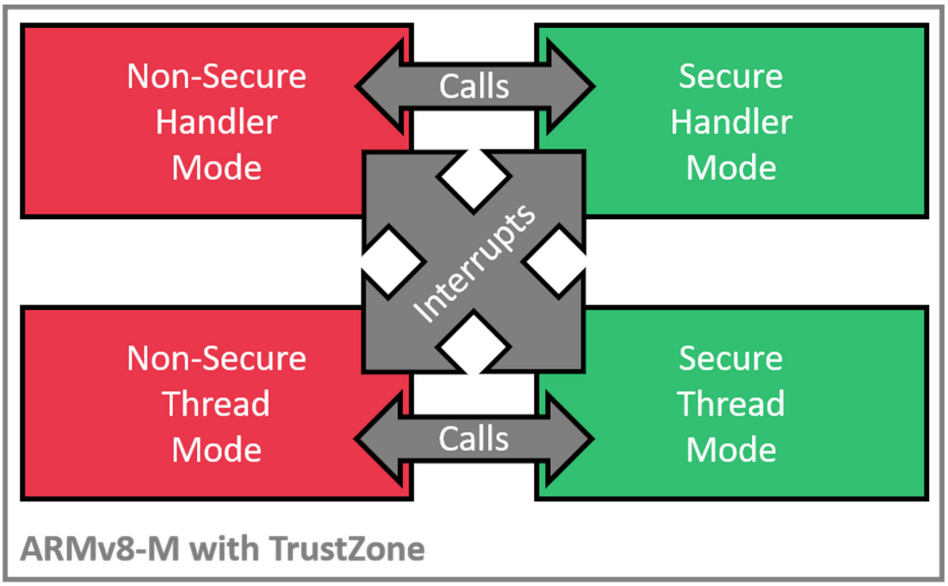
在 SAM L11 Cortex-M23 内核实现中，通过实现定义的属性单元（Implementation Defined Attribution Unit，IDAU）进行安全管理。IDAU 接口负责控制是否可以执行基于当前内核安全状态和指令地址的特定指令。下图给出了系统在允许访问特定存储区之前所执行的内核/调试器访问验证。

图 1-2. IDAU 接口和存储器访问



得益于上述实现，只需使用简单的函数调用或中断处理即可转移到特定的安全状态，具体如下图所示。这样便不会产生任何代码和执行开销，从而实现高效调用。

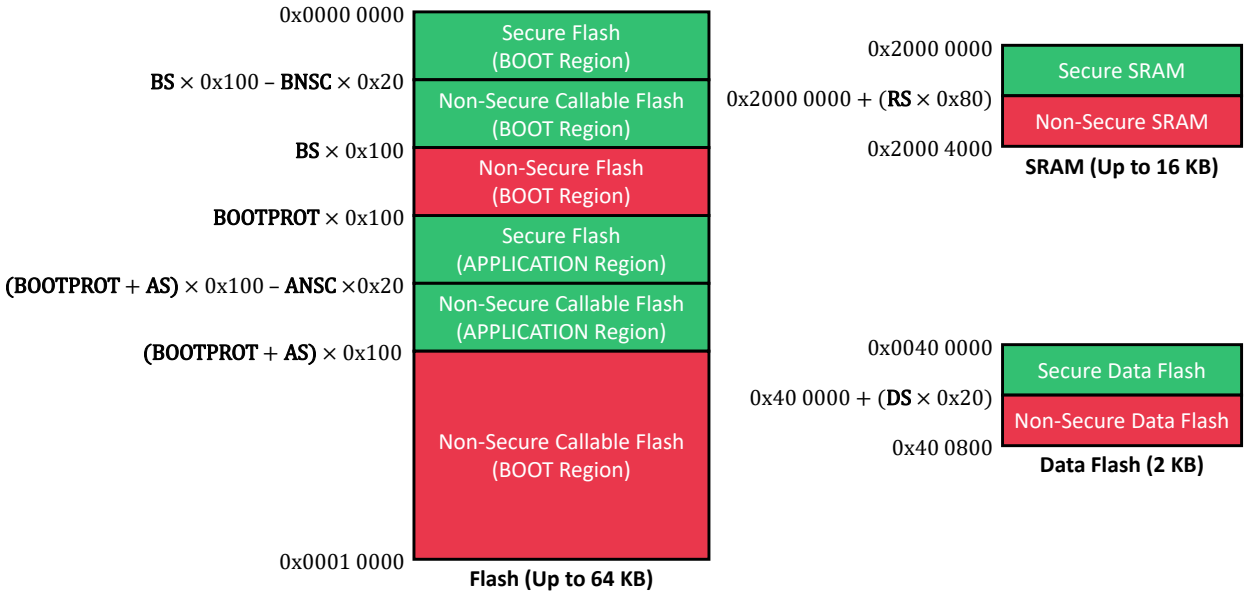
图 1-3. ARMv8-M 的 TrustZone 状态切换



1.1.1 存储器安全属性

为了区分并隔离安全代码与非安全代码，SAM L11 存储器划分为 10 个存储区，如下图所示。每个区域的大小可使用专用 NVM 熔丝（例如 BS、BNSC、BOOTPROT、AS、ANSC、DS 和 RS）进行配置。

图 1-4. SAM L11 存储区



每个存储区会在硬件中预配置为下列其中一种属性：

- **非安全（NS）**：非安全地址用于存储器和外设，器件上运行的所有软件都可以访问。
- **安全（S）**：安全地址也用于存储器和外设，但只有安全软件可以访问。
- **非安全可调用（NSC）**：NSC 是一种特殊的安全存储单元，支持软件从非安全状态切换为安全状态。

每个区域的安全属性将定义该区域中存储的代码的安全状态。

1.1.2 安全和非安全函数调用机制

为了防止从非安全状态下访问安全代码和数据，安全代码必须同时满足多项要求。为此，需要通过 MCU 架构、软件架构与工具链配置来协同实现。

在内核级，ARMv8-M 器件有一组专用的安全指令，可用于在 CPU 安全状态切换期间保存和保护安全寄存器值。

- **安全网关（Secure Gateway，SG）**：用于在执行安全入口点的第一条指令时从非安全状态切换为安全状态。
- **转移并交换到非安全状态（Branch with eXchange to Non-Secure State，BXNS）**：供安全软件用于转移或返回到非安全程序。
- **通过链接转移并交换到非安全状态（Branch with Link and eXchange to Non-Secure State，BLXNS）**：供安全软件用于调用非安全函数。

在工具链级，必须使用 Arm 提供的“C”语言扩展（CMSE）来确保使用 ARMv8-M 安全指令。

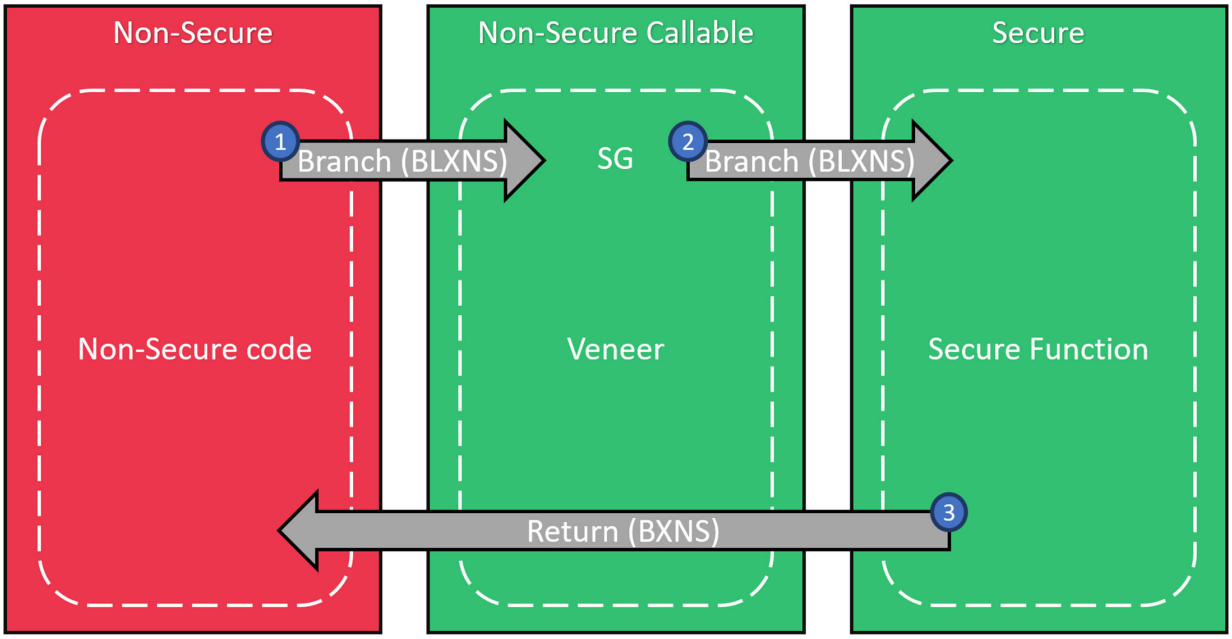
在软件架构级，必须使用特定安全和非安全函数调用机制来确保安全性，具体如以下章节所述：

1.1.2.1 非安全可调用 API

使用 ARMv8-M 的 TrustZone 技术时，应用程序开发人员可定义一组非安全可调用 API，用于从非安全区域访问安全代码。这些 API 称为安全网关（SG）或模板，负责 CPU 安全状态切换，并且能够将安全入口点与安全代码的其余部分隔离，从而限制非安全状态下能够访问的代码量。

SG 应置于 NSC 存储区，以确保只能在 CPU 处于非安全状态时执行。安全代码的其余部分应置于安全存储区，以确保无法在 CPU 处于非安全状态时访问，具体请参见下图。

图 1-5. 非安全可调用 API 机制



若要使用非安全可调用 API，需使用特定 Cortex-M23 指令来确保内核安全状态切换期间的安全性。只有入口点的第一条指令为 SG 且位于非安全可调用存储单元时，才允许从非安全到安全软件入口点的直接 API 函数调用。要转移到非安全代码，还需要使用特殊指令（BXNS 和 BLXNS）。

以下代码通过结合使用 Arm GCC 工具链与“C”语言扩展（CMSE）对安全函数及其 SG API 声明和定义进行了说明。

Veneer.h：

```
/* 非安全可调用函数 */
extern int nsc_func1(int x);
```

Veneer.c（链接到器件的 NSC 存储区中）：

```
/* 非安全可调用（入口）函数 */
int __attribute__((cmse_nonsecure_entry)) nsc_func1(int x)
{
    return secure_func1(x);
}
```

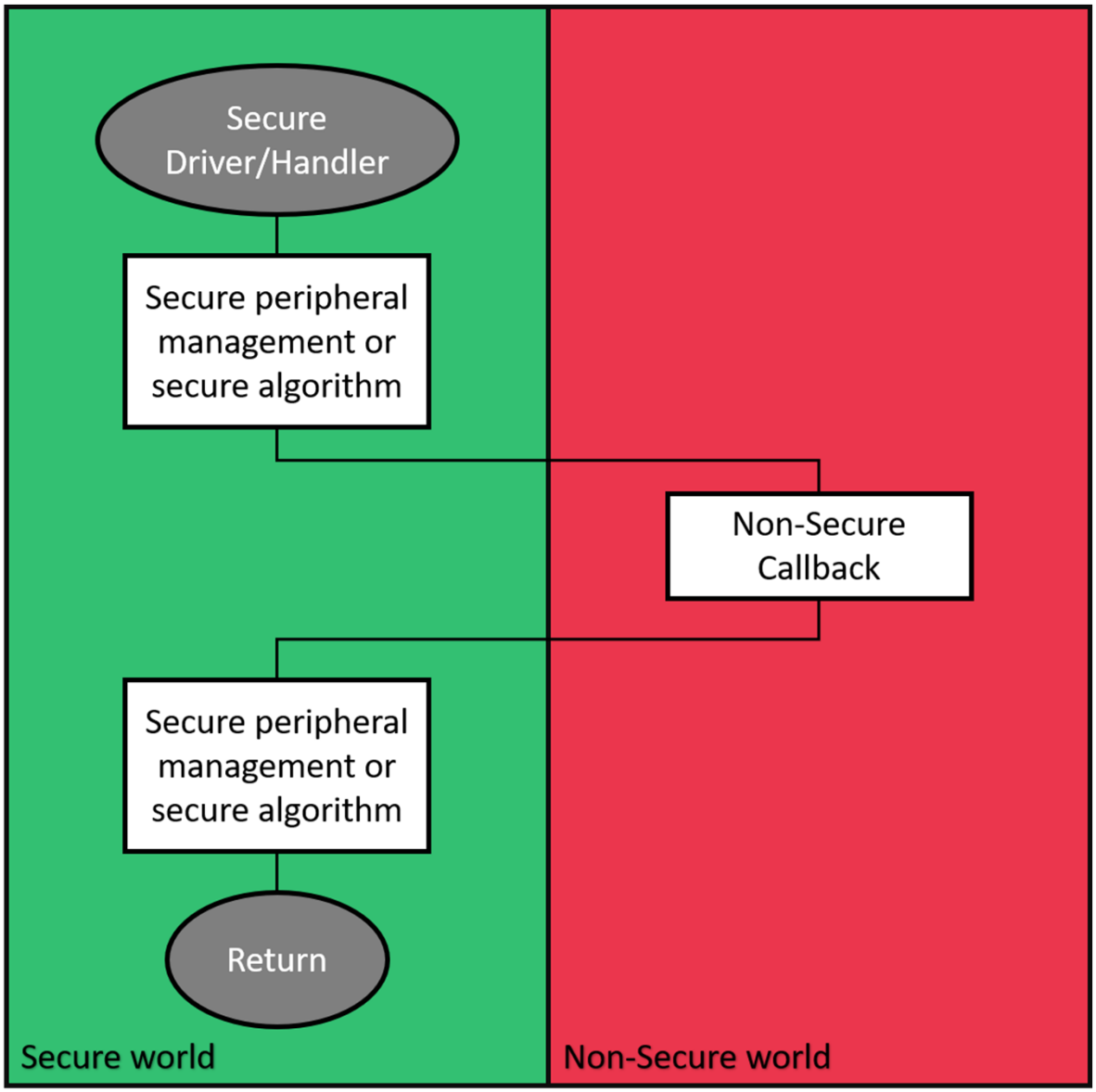
Secure_function.c（链接到器件的安全存储区中）：

```
int secure_func1(int x)
{
    return x + 3;
}
```

1.1.2.2 非安全软件回调

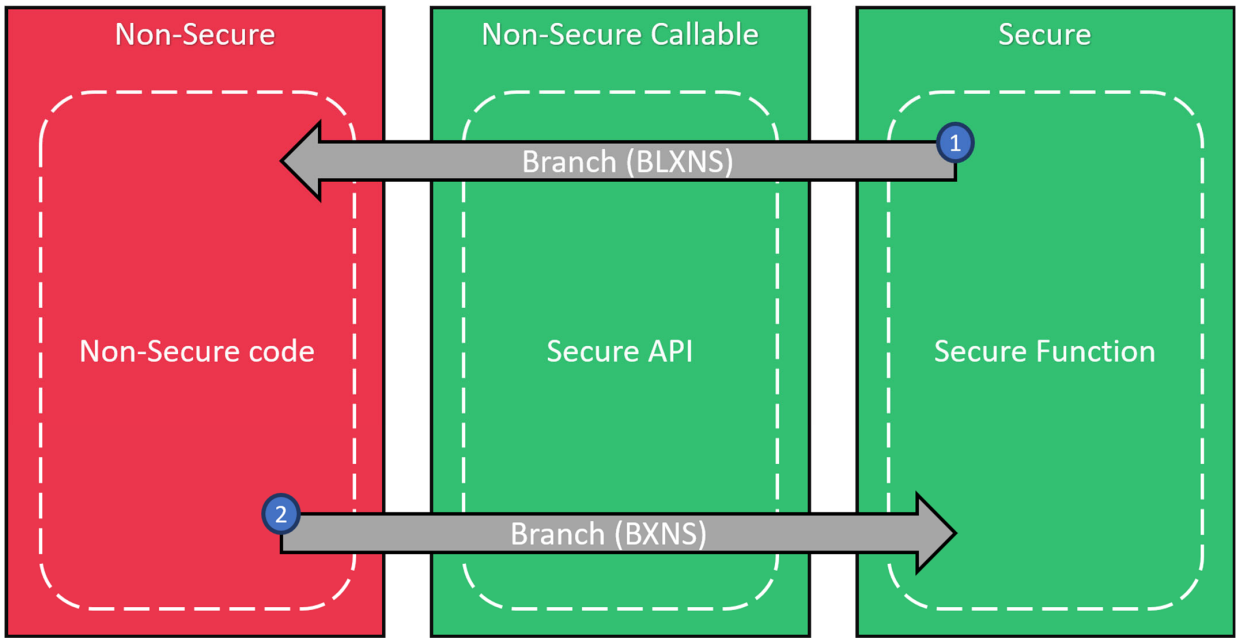
安全代码可以定义并使用软件回调从非安全区域执行函数，这是因为安全代码与非安全代码分别位于单独的可执行文件中。下图所示为软件回调方法。

图 1-6. 非安全软件回调流程图



可以使用 BLXNS 指令管理回调函数。下图所示为非安全回调机制：

图 1-7. 非安全软件回调机制

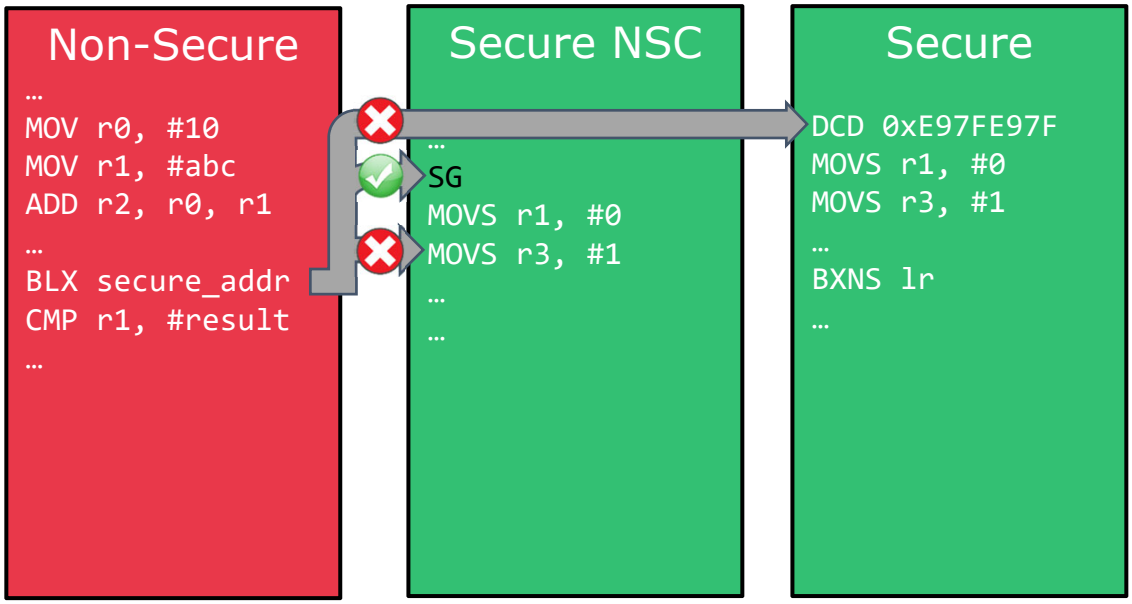


注：非安全软件回调通过指向非安全代码单元的指针来定义。务必在安全应用程序中检查指针是否正确，误用指针会产生安全漏洞，进而导致非安全代码也可以执行任何安全函数。为了克服这一缺点，我们提供了一组基于全新 Cortex-M23 测试目标（Test Target，TT）指令的 CMSE 函数。

1.1.2.3 安全状态和调用不匹配

如果尝试从非安全代码访问安全区域，或者执行的代码与系统安全状态之间不匹配，则会导致硬故障异常，具体如下图所示。

图 1-8. 安全状态和调用不匹配



1.1.3 安全和非安全中断处理

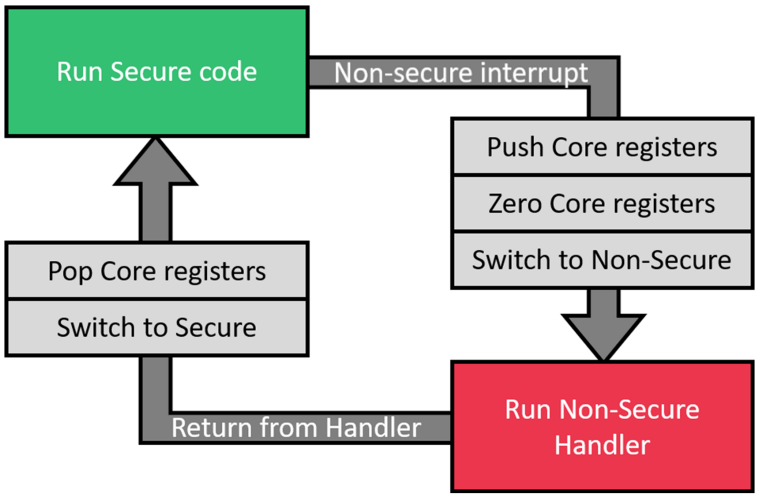
Cortex-M23（ARMv8-M 架构）使用的异常堆栈机制与 ARMv7-M 架构相同，其中内核寄存器的子集自动存储到堆栈中（硬件现场保护）。这样可以立即执行中断处理程序，而无需通过软件执行现场保护。ARMv8-M 将该机制扩展为基于两个不同的堆栈指针（安全堆栈指针和非安全堆栈指针）提供增强的安全性。

根据嵌套向量中断控制器（Nested Vector Interrupt Controller，NVIC）中配置的优先级设置，安全代码执行可以中断非安全代码执行，非安全代码执行也可以中断安全代码执行。内核级 NVIC 寄存器会进行复制。这样便有两个向量表定义，一个用于安全，另一个用于非安全。

当产品启动时，所有中断默认映射到安全区域（安全向量表）。安全区域中可访问的特定 CMSIS 函数将每个中断向量分配给非安全处理程序（在非安全向量表中声明）。

如下图所示，如果高优先级非安全中断到达时正在运行安全代码，则内核会将安全寄存器内容全部压入专用安全堆栈中。寄存器随后自动清零以防止读取任何信息，而内核执行非安全异常处理程序。当非安全处理程序执行完毕时，硬件将自动从安全堆栈中恢复所有寄存器内容。该机制通过硬件管理，无需任何软件干预。这样可以从运行安全代码安全地切换到运行非安全中断处理程序，然后再恢复运行安全代码。

图 1-9. Cortex-M23 中断机制



1.2 外设安全属性

SAM L11 系列器件将 TrustZone 的概念扩展到其集成的外设，并且能够将特定外设分配给安全区域和非安全区域。此外，SAM L11 还内置可在安全应用程序与非安全应用程序之间共享资源的外设，称为混合安全外设。每个外设的安全属性通过外设访问控制器（Peripheral Access Controller，PAC）进行管理。

注：IDAU 外设始终安全，器件服务单元（Device Service Unit，DSU）外设始终非安全。有关更多信息，请参见《SAM L10/L11 系列数据手册》。

1.2.1 安全和非安全外设

如下图所示，PAC 控制器内置一组寄存器，用于定义系统中每个集成外设的安全属性。这些寄存器在器件启动时由 ROM 代码配置，将根据用户行（UROW）熔丝中存储的用户配置设置 PAC.NONSECx 寄存器。

图 1-10. PAC.NONSECx 寄存器说明

NONSECA	7:0	GCLK	SUPC	OSC32KCTRL	OSCCTRL	RSTC	MCLK	PM	PAC
	15:8			AC	PORT	FREQM	EIC	RTC	WDT
	23:16								
	31:24								
NONSECB	7:0				HMATRIXHS	DMAC	NVMCTRL	DSU	IDAU
	15:8								
	23:16								
	31:24								
NONSECC	7:0	ADC	TC2	TC1	TC0	SERCOM2	SERCOM1	SERCOM0	EVSYS
	15:8			TRAM	OPAMP	CCL	TRNG	PTC	DAC
	23:16								
	31:24								

重要：在应用程序运行期间，无法通过访问 PAC.NONSECx 寄存器更改外设安全属性。任何更改都必须使用用户行熔丝来完成，并且需要复位 SAM L11 器件。应用程序可读取 PAC.NONSECx 寄存器来获取集成外设的当前属性。

外设可按照其 PAC 安全属性及其内部安全分区能力（标准/混合安全）分为两组：

- 非安全外设：标准外设在 PAC 中配置为非安全。整个外设的安全属性通过将相关 NONSECx 熔丝定义设置为 1 来定义。授予对外设的安全和非安全访问权限。
- 安全外设：标准外设在 PAC 中配置为安全。整个外设的安全属性通过将相关 NONSECx 熔丝定义设置为 0 来定义。授予对外设的安全访问权限，非安全访问将被丢弃（忽略写操作，读为 0x0）并触发 PAC 错误。

将外设分配给安全区域时，仅授予对其寄存器的安全访问权限，并且只能在安全区域管理中断处理。

1.2.2 混合安全集成外设

SAM L11 内置 5 个混合安全外设，这些外设的部分内部资源可在安全区域与非安全区域之间共享。下面完整列出了 SAM L11 混合安全外设及其共享资源：

- 外设访问控制器（PAC）：管理外设安全属性（安全或非安全）。
- 非易失性存储器控制器（NVMCTRL）：处理安全和非安全闪存区域编程。
- I/O 引脚控制器（PORT）：支持将每个 I/O 单独分配给安全或非安全应用程序。
- 外部中断控制器（External Interrupt Controller，EIC）：支持将每个外部中断单独分配给安全或非安全应用程序。
- 事件系统（EVSYS）：支持将每个事件通道单独分配给安全或非安全应用程序。

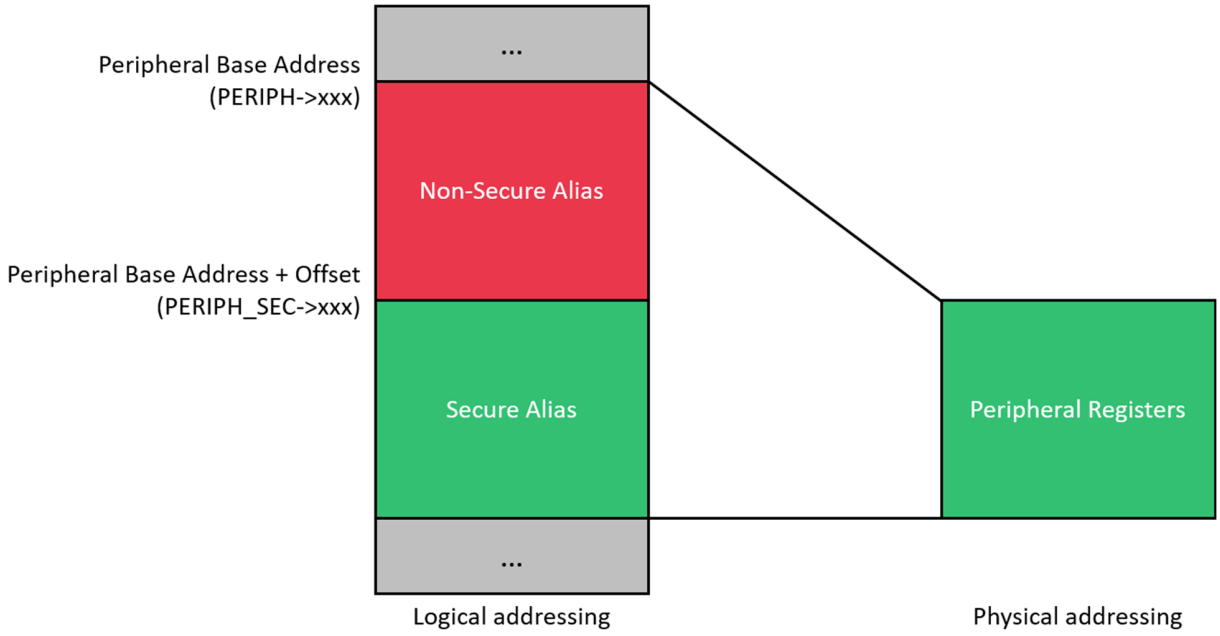
混合安全外设共享其内部资源的能力取决于该外设在 PAC 外设中的安全属性（PAC 安全或 PAC 非安全）。

- 当混合安全外设为安全（NONSECx 熔丝设置为 0）时，安全区域可使用专用寄存器将外设的内部资源分配给非安全区域。
- 当混合安全外设为非安全（NONSECx 熔丝设置为 1）时，外设的行为类似于标准非安全外设。授予对外设寄存器的安全和非安全访问权限。

1.2.2.1 混合安全外设（PAC 安全）

当混合安全外设为 PAC 安全（相关 PAC NONSECx 熔丝设置为 0）时，外设寄存器会进行分区，并且可通过两个不同的存储器别名进行访问，如下图所示。

图 1-11. PAC 安全的混合安全外设寄存器寻址



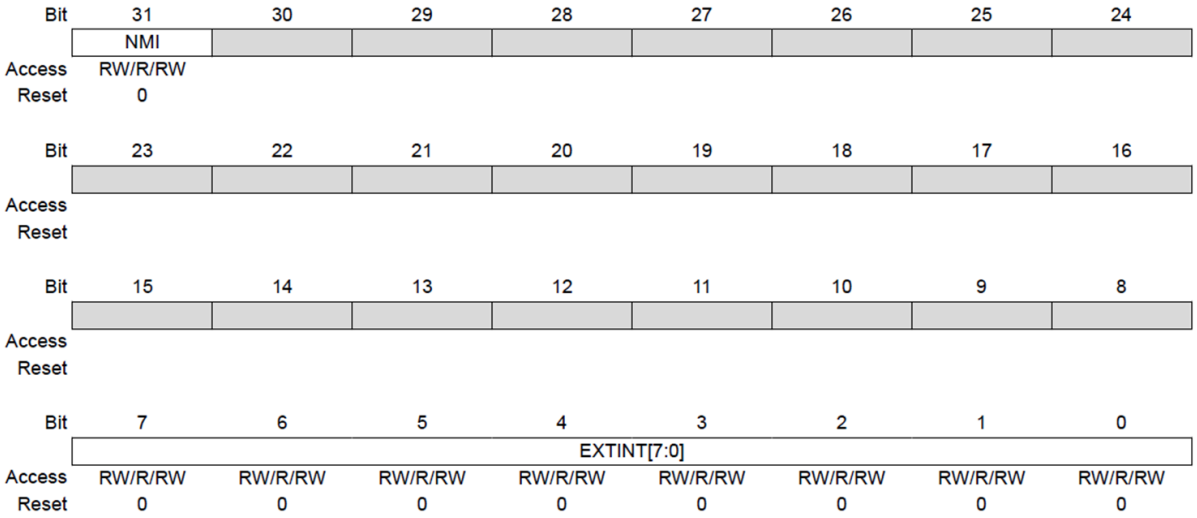
安全区域随后可使用 NONSEC 寄存器单独使能对内部外设资源的非安全访问，如下图（针对外部中断控制器）所示。

图 1-12. 外部中断控制器 NONSEC 寄存器

Name: NONSEC
Offset: 0x40
Reset: 0x00000000
Property: PAC Write-Protection, Write-Secure

This register allows to set the NMI or external interrupt control and status registers in non-secure mode, individually per interrupt pin.

Important: This register is only available for **SAM L11** and has no effect for **SAM L10**.



Bit	31	30	29	28	27	26	25	24
	NMI							
Access	RW/R/RW							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	EXTINT[7:0]							
Access	RW/R/RW	RW/R/RW	RW/R/RW	RW/R/RW	RW/R/RW	RW/R/RW	RW/R/RW	RW/R/RW
Reset	0	0	0	0	0	0	0	0

NONSEC 寄存器的内容只能由安全区域通过外设寄存器安全别名（PERIPH_SEC.NONSEC）修改。

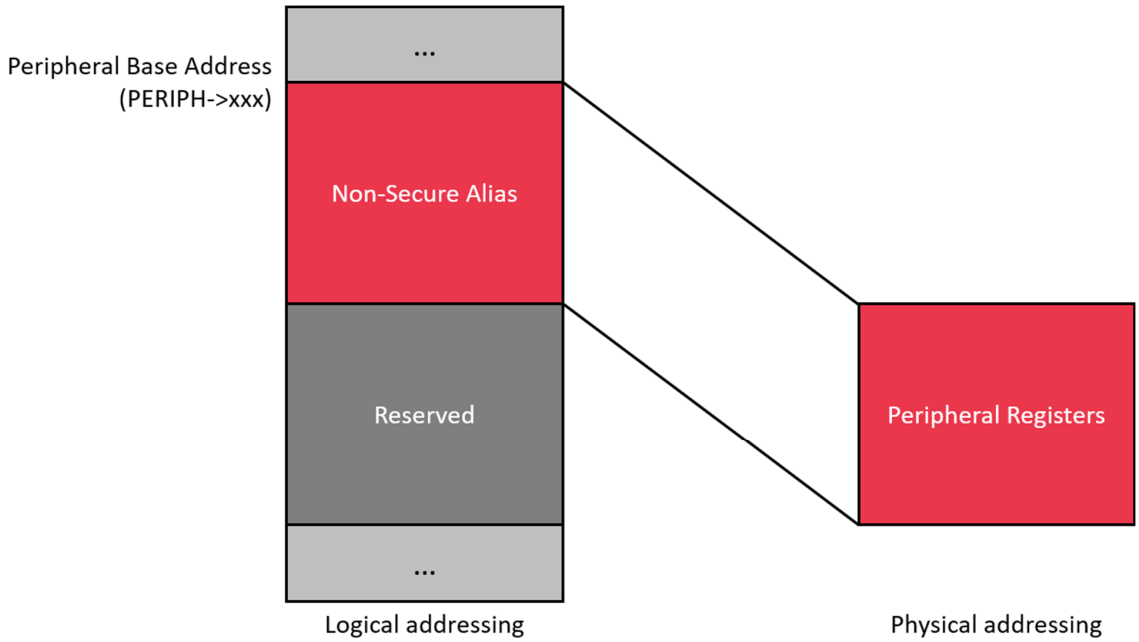
通过设置 NONSEC 寄存器中的特定内部功能位域，可以访问外设非安全别名中与该功能相关的不同位域。

1.2.2.2 混合安全外设（PAC 非安全）

当混合安全外设为 PAC 非安全（相关 NONSECx 熔丝设置为 1）时，外设的行为类似于标准非安全外设。

授予对外设寄存器的安全和非安全访问权限。外设寄存器映射如下图所示：

图 1-13. PAC 非安全混合安全外设寄存器寻址



在应用程序级管理 PAC 非安全的混合安全外设与管理标准非安全外设类似。

1.3 调试访问级别（DAL）和全片擦除

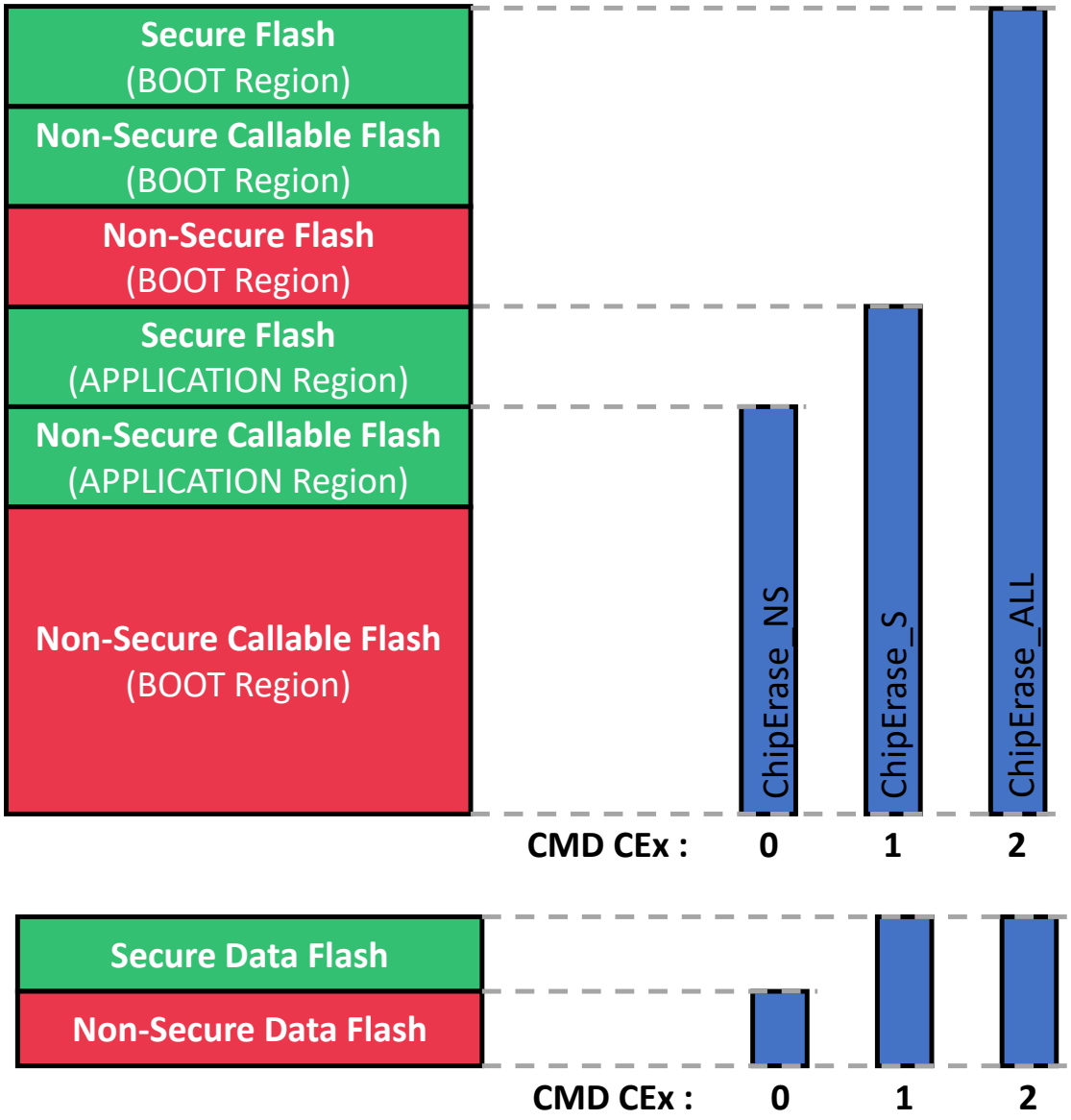
SAM L11 具有以下可配置调试访问级别（Debug Access Level，DAL），用于限制对系统中安全和非安全资源的编程和调试访问。

- **DAL2**：存储器和外设访问不受限制的调试访问
- **DAL1**：仅限非安全存储区访问，禁止安全存储区访问
- **DAL0**：未授予任何访问权限，但调试器使用引导 ROM 交互模式时除外

注：有关引导交互模式的更多信息，请参见《SAM L11 数据手册》（DS60001513E_CN）中的“引导 ROM”一章。

调试访问级别与三个受密钥保护的全片擦除命令结合使用，可提供三种级别的非易失性存储器擦除粒度，如下图所示。

图 1-14. 全片擦除命令



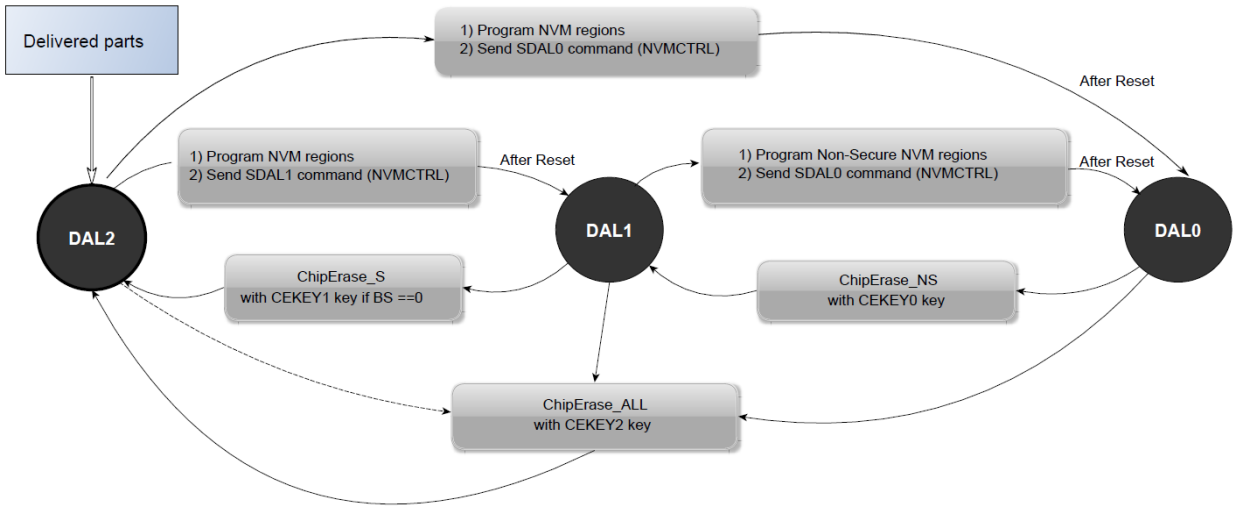
全片擦除命令保护密钥通过 BOCOR 位域来配置，如下图所示。

图 1-15. SAM L11 可配置全片擦除密钥熔丝

Offset	Bit Pos.	Name			
0x00	7:0	Reserved			
0x01	15:8	BS			
0x02	23:16	Reserved	BNSC		
0x03	31:24	BOOTOPT			
0x04	39:32	BOOTPROT			
0x05	47:40	Reserved			
0x06	55:48	Reserved		BCREN	BCWEN
0x07	63:56	Reserved			
0x08-0x0B	95:64	BOCORCRC			
0x0C-0x0F	127:96	ROMVERSION			
0x10-0x1F	255:128	CEKEY0			
0x20-0x2F	383:256	CEKEY1			
0x30-0x3F	511:384	CEKEY2			
0x40-0x4F	639:512	CRCKEY			
0x50-0x6F	895:640	BOOTKEY			
0x70-0xDF	1791:896	Reserved			
0xE0-0xFF	2047:1792	BOCORHASH			

使用不同的全片擦除命令可以在不影响代码安全性的条件下提高 DAL 级别。因此，在切换为较高的 DAL 级别前，应先擦除代码，如下图所示。

图 1-16. SAM L11 DAL 和全片擦除机制



Microchip Studio 7 内的器件编程实用程序提供了最简单的方式来设置 DAL 命令和全片擦除命令，并且还可用于访问器件熔丝，如以下各图所示。

图 1-17. Microchip Studio 7 器件编程期间的全片擦除命令

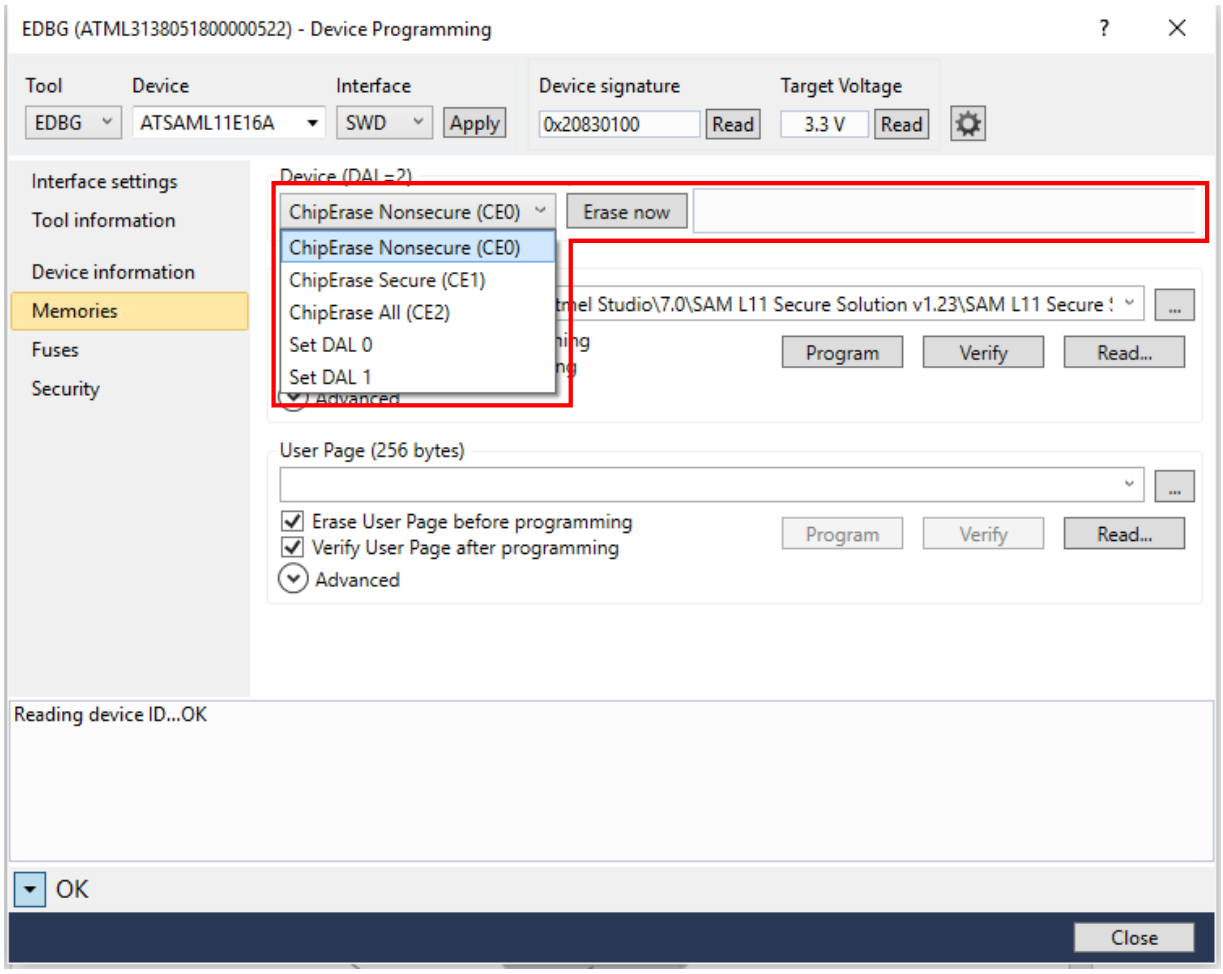
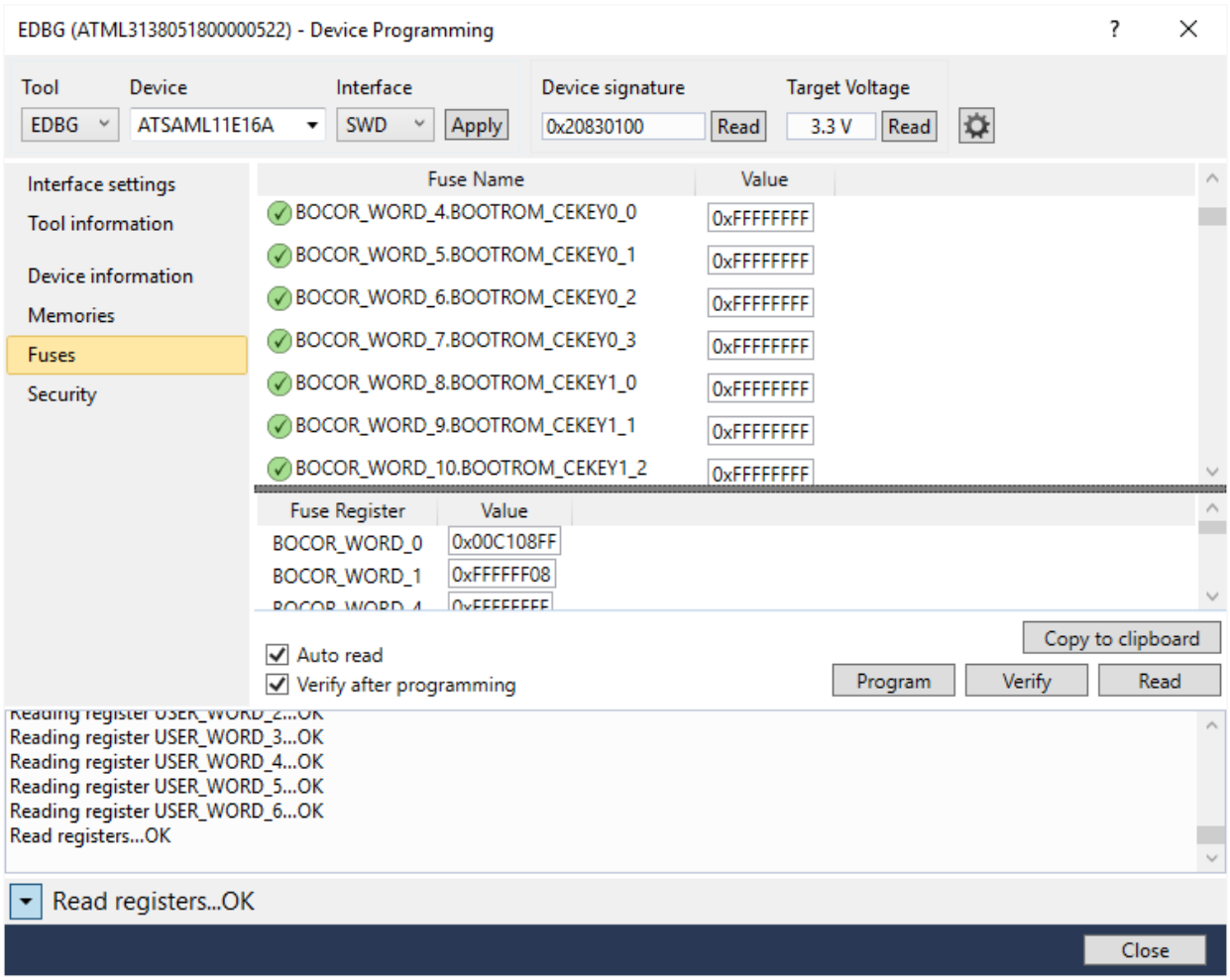


图 1-18. Microchip Studio 7 器件编程期间的全片擦除密钥熔丝设置

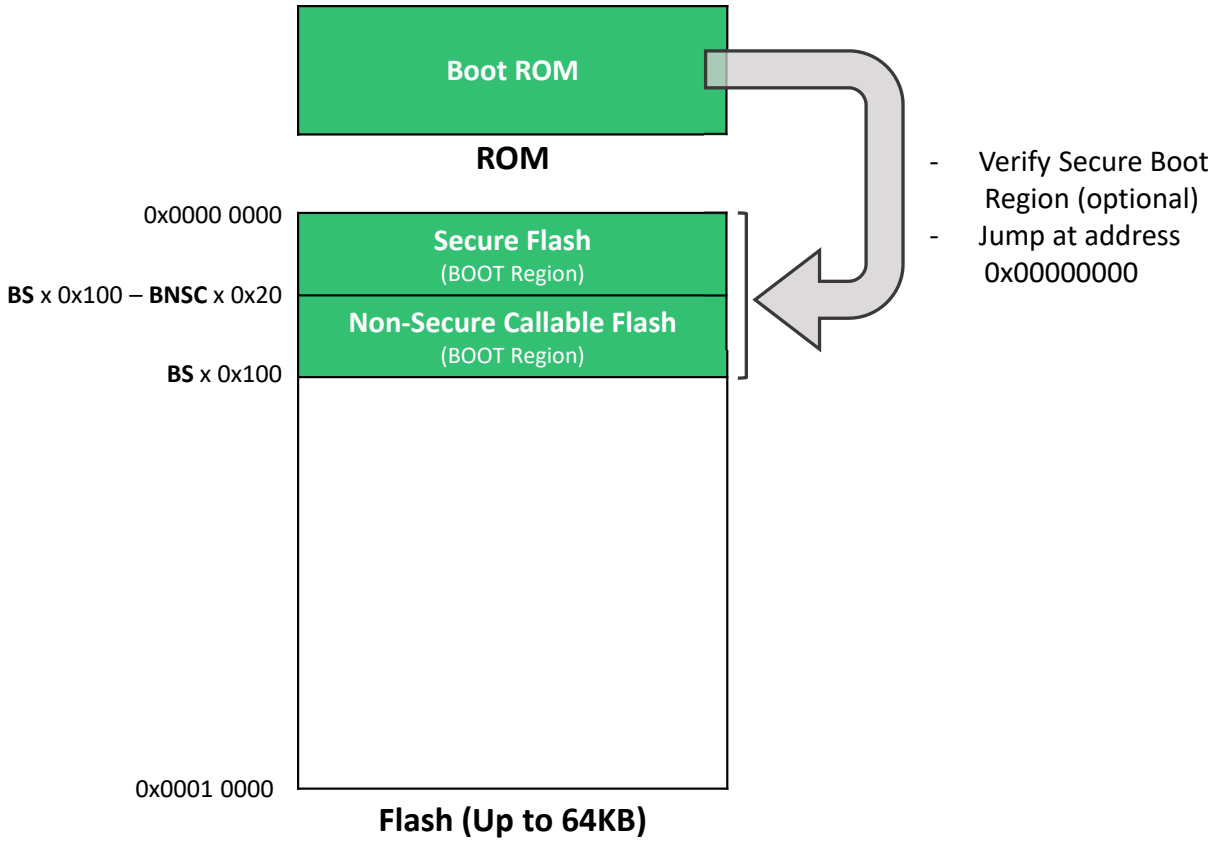


1.4 安全引导

SAM L11 引导 ROM 始终在产品启动阶段执行。该软件通过 ROM 编码到器件中，用户不能跳过。引导 ROM 可根据引导配置行（BOCOR）熔丝设置了解系统中是否定义了安全引导区域。

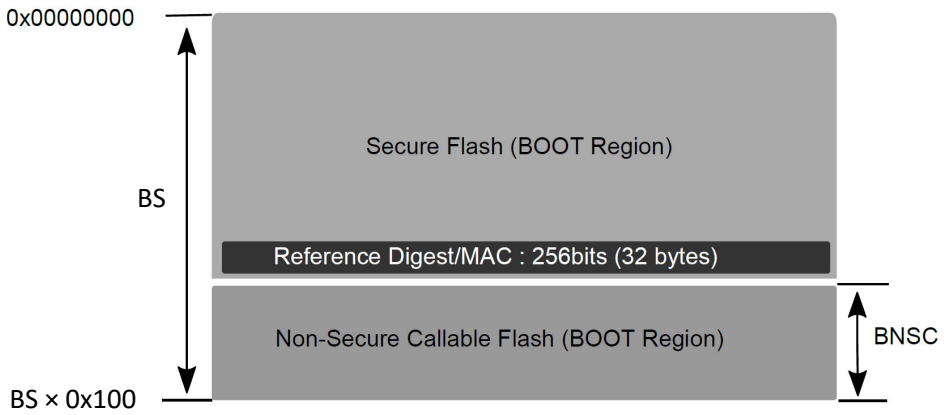
引导 ROM 可在执行前进行完整性检查（SHA-256）或者验证（SHA-256 + BOOTKEY）安全引导区域中存储的固件。该验证机制是在部署和执行安全固件期间确保系统可信根而需考虑的关键要素。下图说明了安全引导过程的 BS（安全 + NSC 引导子区域）验证。

图 1-19. 安全引导过程的 BS 验证



要验证器件闪存 BS 存储器段中存储的安全自举程序代码，ROM 代码需使用加密加速器（CRYA）计算闪存 BS 区域的哈希值，并将其与器件安全闪存（引导区域）存储器段中存储的参考哈希值（256 位/32 字节）进行比较。该参考哈希值（256 位）必须存储在安全闪存（引导区域）的最后 256 位中，如下图所示。

图 1-20. 引导安全参考哈希值位置



如果验证结果等于参考哈希值，则引导 ROM 将开始执行安全自举程序。如果值不匹配，器件将进入无限复位循环，以防止执行闪存代码。此时，只能通过 ChipErase_ALL 命令使器件从该状态中恢复。ChipErase_ALL 命令会擦除存储器的全部内容并将熔丝复位为其出厂设置。

安全引导过程配置中使用以下熔丝：

- **BOOTPROT、BS 和 BSNC：**定义产品闪存中引导段的配置。安全、非安全和非安全可调用引导段的大小可根据应用需求自定义。这些熔丝用于产品 IDAU 中的安全存储器分配，以及完整性和身份验证机制（在 BOOTOPT 熔丝中配置时）。更改熔丝设置需要考虑复位器件，因为只有引导 ROM 能够更改 IDAU 设置。
- **BOOTOPT：**定义要执行的验证的类型。

表 1-1. SAM L11 安全引导验证方法

BOOTOPT	BOOTPROT 区域 验证方法	BOCOR 行 验证方法
0	禁止安全引导	
1	SHA-256（完整性检查）	
2 或 3	SHA-256 + BOOTKEY(1)（身份验证检查）	

注：

1. BOOTKEY 在 BOCOR 行中定义。
2. 使用安全引导身份验证功能会影响产品启动时间。请参见《SAM L10/L11 系列数据手册》（DS60001513E_CN）。

注：使用安全引导身份验证功能会影响产品启动时间。有关更多信息，请参见《SAM L10/L11 数据手册》（DS60001513E_CN）。

BOOTKEY：用于身份验证机制的 256 位 BOOTKEY。

下图突出显示了用于配置安全引导过程的熔丝。

图 1-21. 安全引导过程熔丝

Bit Pos.	Name	Usage	Factory Setting	Related Peripheral Register
7:0	Reserved	Reserved	Reserved	Reserved
15:8	BS	Secure Flash (BS region) Size = BS*0x100 (2)	0x00	IDAU.SCFGB
21:16	BNSC	Non-Secure Callable Flash (BOOT region) Size = BNSC*0x20	0x00	IDAU.SCFGB
23:22	Reserved	Reserved	Reserved	Reserved
25:24	BOOTOPT	Boot Option	0x0	Boot ROM
31:26	Reserved	Reserved	Reserved	Reserved
39:32	BOOTPROT	Boot Protection size = BOOTPROT*0x100	0x00	IDAU.SCFGB
47:40	Reserved	Reserved	Reserved	Reserved
48	BCWEN	Boot Configuration Write Enable	0x1	NVMCTRL.SCFGB
49	BCREN	Boot Configuration Read Enable	0x1	NVMCTRL.SCFGB
63:50	Reserved	Reserved	Reserved	Reserved
95:64	BOCORCRC	Boot Configuration CRC for bit 63:0	0xDDE78140(1)	Boot ROM
127:96	Reserved	Reserved	Reserved	Reserved
255:128	CEKEY0	Chip Erase Key 0	All 1s	Boot ROM
383:256	CEKEY1	Chip Erase Key 1	All 1s	Boot ROM
511:384	CEKEY2	Chip Erase Key 2	All 1s	Boot ROM
639:512	CRCKEY	CRC Key	All 1s	Boot ROM
895:640	BOOTKEY	Secure Boot Key	All 1s	Boot ROM
1791:896	Reserved	Reserved	Reserved	Reserved
2047:1792	BOCORHASH	Boot Configuration Row Hash	All 1s	Boot ROM

2. SAM L11 应用程序开发（开发人员 A 和开发人员 B）

系统 DAL 和全片擦除与 Cortex-M 架构的 TrustZone 技术结合使用，因此开发人员可采用以下开发和部署方法：

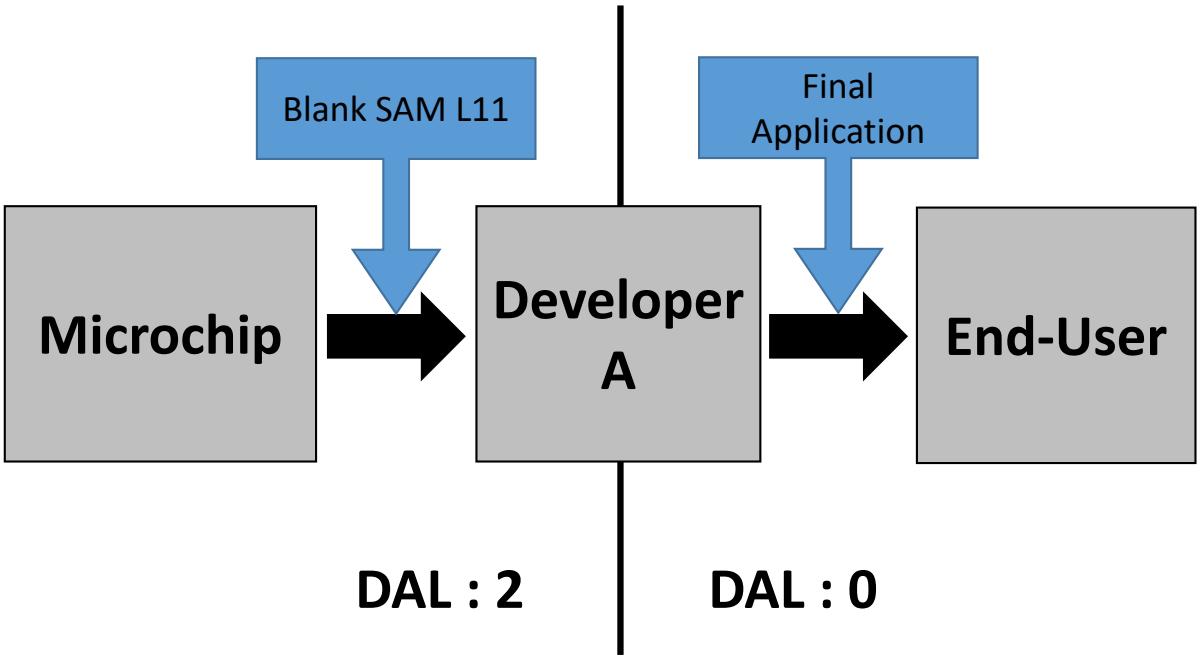
- 单开发人员方法（开发人员 A）
- 双开发人员方法（开发人员 A + 开发人员 B）

Microchip Studio 7 集成开发平台提供了一整套高级功能来加快 SAM L11 应用程序的开发过程。以下章节介绍了开发人员 A 与开发人员 B 创建和定制应用程序的方法。

2.1 单开发人员方法

在单开发人员方法中，开发人员（开发人员 A）负责开发和部署安全和非安全代码。开发人员 A 的应用程序可使用 DAL0 进行保护。下图对基于 SAM L11 的单开发人员方法进行了说明。

图 2-1. 单开发人员方法

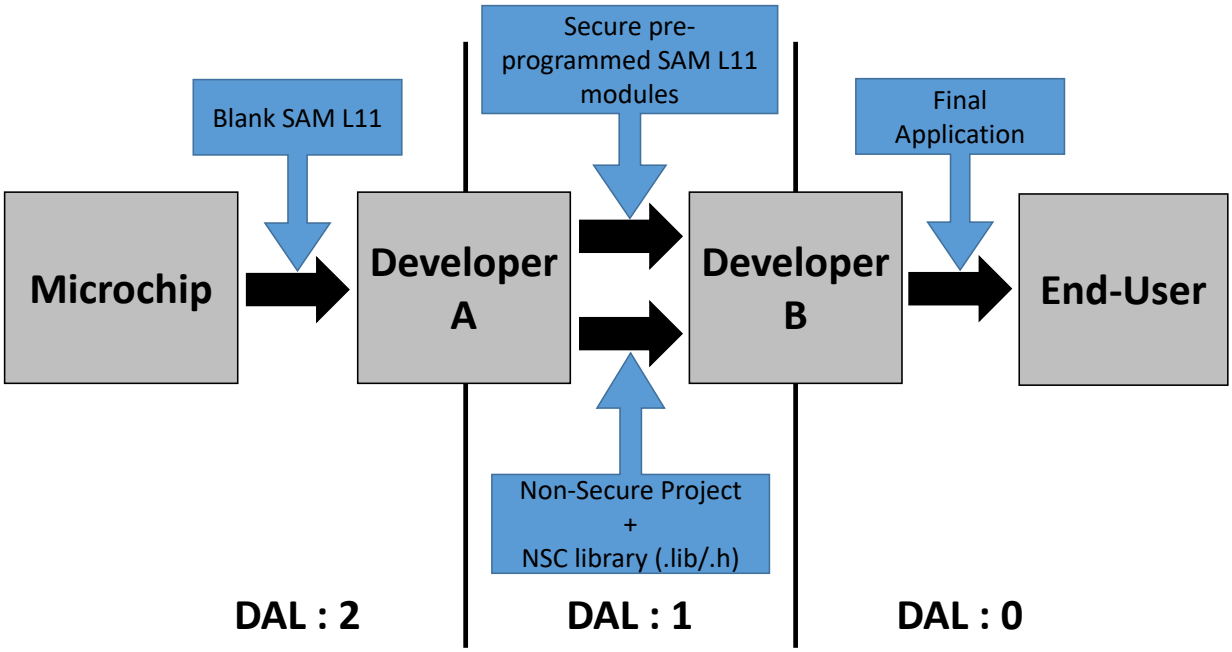


2.2 双开发人员方法

在该方法中，第一个开发人员（开发人员 A）负责开发安全应用程序及其相关的非安全可调用库（`.lib/.h`），以及向第二个开发人员（开发人员 B）提供预定义的链接器文件。该安全应用程序随后装入 SAM L11 闪存，并使用设置的 DAL1 命令进行保护以防止进一步访问器件的安全存储区。

然后，第二个开发人员（开发人员 B）将基于预编程的 SAM L11 着手开发，但被限制访问安全资源（仅限调用非安全 API）。为此，开发人员 B 将使用开发人员 A 提供的链接器文件和 NSC 库。下图对基于 SAM L11 的双开发人员方法进行了说明。

图 2-2. 双开发人员方法

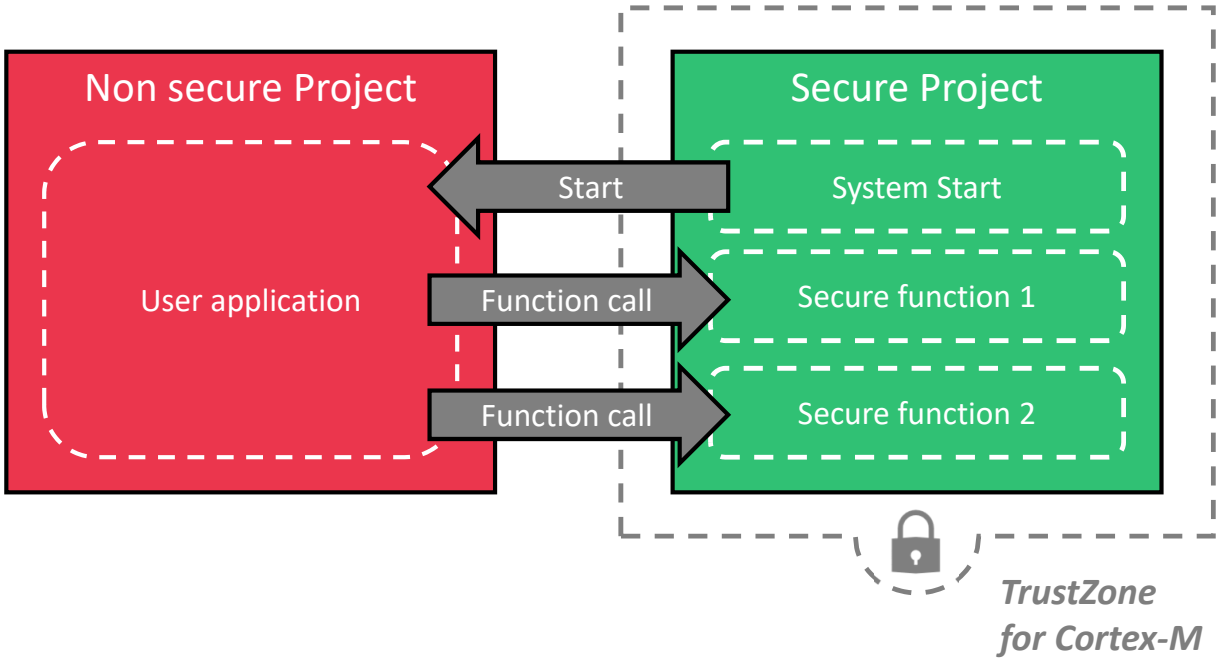


以下章节介绍了开发人员 A 与开发人员 B 执行的应用程序开发和部署过程。

2.3 开发安全解决方案（开发人员 A）

为了帮助开发人员 A 基于 SAM L11 着手开发（无论是单开发人员方法还是双开发人员方法），Microchip Studio 7 提供了预配置的安全解决方案模板，该模板对基本安全和非安全应用程序执行进行了说明（如下图所示），可用于评估和了解器件中 ARMv8-M 实现的 TrustZone 技术或作为定制解决方案开发的起始点。

图 2-3. 安全解决方案模板概览

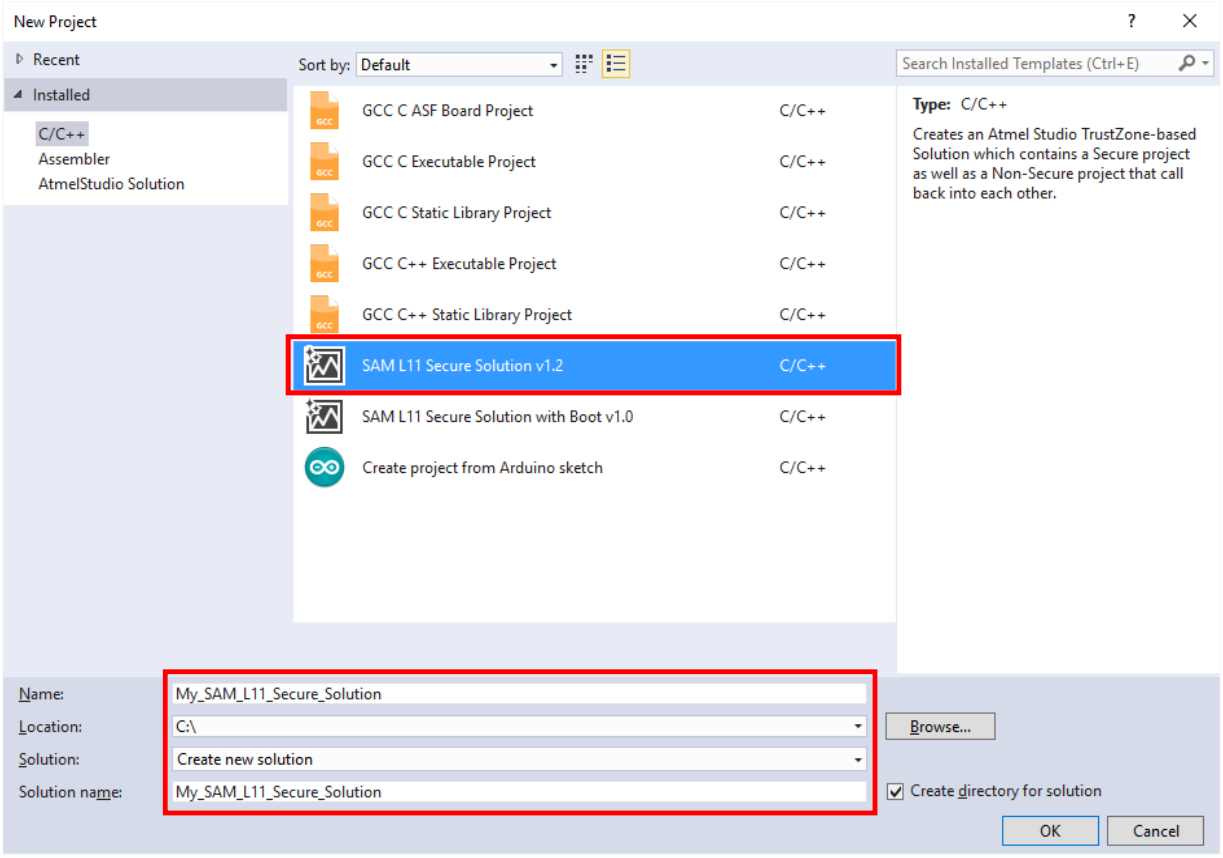


2.3.1 使用 Microchip Studio 安全解决方案模板创建 SAM L11 安全解决方案

要使用 Microchip Studio 7 中提供的预配置模板创建安全解决方案，可按照以下步骤操作：

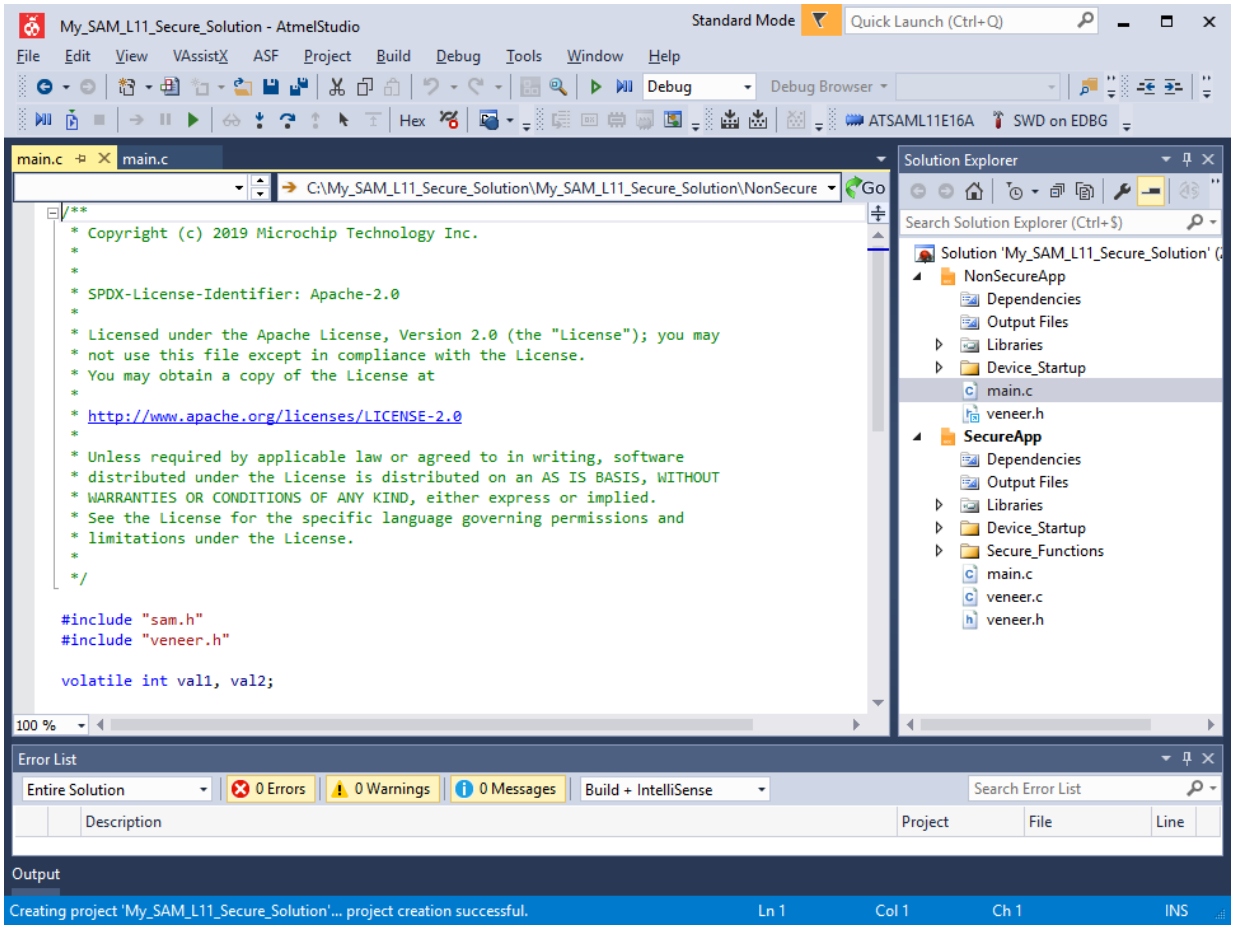
1. 打开 Microchip Studio 7。
2. 选择 *File* > *New* > *Project*（文件 > 新建 > 项目）。
3. 在 New Project（新建项目）窗口中，执行以下操作来创建和配置一个新的解决方案：
   a. 展开 Installed（已安装工具），选择 C/C++。
   b. 选择 SAM L11 安全解决方案。
   c. 在 Name（名称）、Location（位置）、Solution（解决方案）和 Solution name（解决方案名称）中输入详细信息（相关示例见下图）。
   d. 单击 **OK**（确定）。

图 2-4. 在 Microchip Studio 7 下创建 SAM L11 解决方案



创建后，SAM L11 安全解决方案应出现在 Microchip Studio 7 IDE 中，如下所示：

图 2-5. Microchip Studio 7 下的 SAM L11 安全解决方案



2.3.2 安全解决方案模板说明

基于 SAM L11 安全解决方案模板（随 Microchip Studio 7 一起提供）创建的任何解决方案都由预配置的非安全项目和安全项目组成。

为了简化开发过程，与 ARMv8-M 实现的 TrustZone 技术相关的所有配置方面均已实现。以下章节介绍了模板内容，以及根据应用需求定制解决方案时需要修改的关键要素。

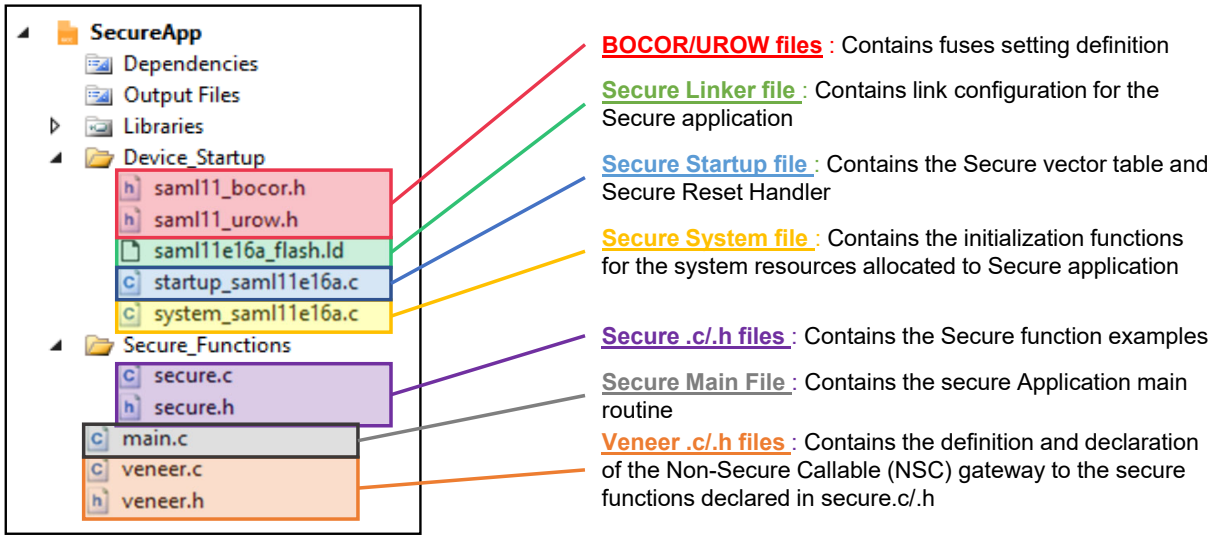
2.3.2.1 安全项目说明

SAM L11 安全解决方案模板中包含的安全项目旨在为基于 SAM L11 的安全代码开发提供预配置的开发库。该安全项目预配置为说明基于 SAM L11 的标准安全应用程序的以下适用方面：

- 器件资源的安全和非安全区域属性（熔丝设置）
- 系统安全的初始化
- 安全函数示例的定义和声明
- 非安全区域的安全网关（模板）的定义和声明
- 对非安全应用程序的安全调用

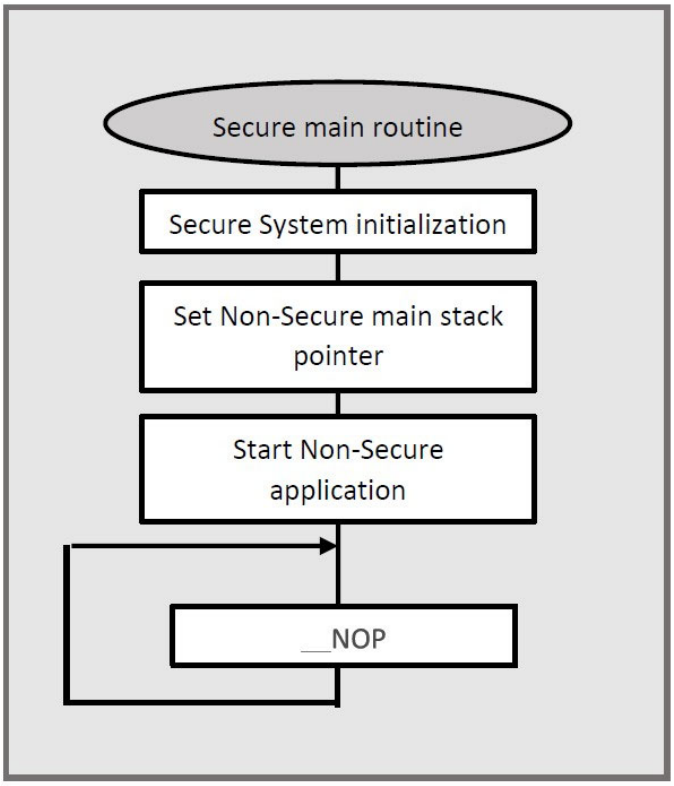
下图对预配置安全项目的文件架构进行了说明：

图 2-6. 安全项目架构



下图对预配置安全项目的主程序进行了说明：

图 2-7. 安全项目主程序流程图



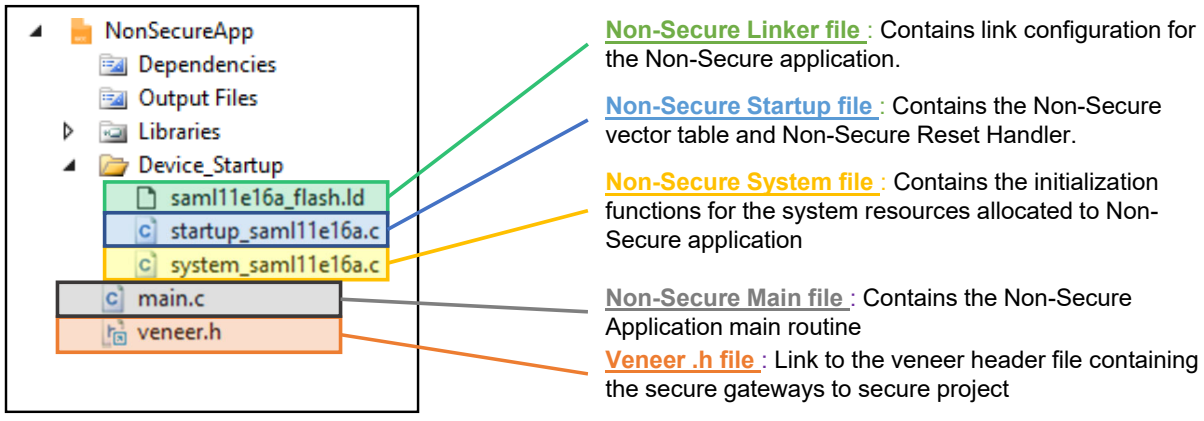
必须将该安全 `main.c` 文件用作任何安全应用程序开发的起始点。

注：提供的 `system_init` 函数为空，因此系统以 4 MHz 运行（复位状态）。该函数应根据安全和非安全应用要求进行定制。

2.3.2.2 非安全项目说明

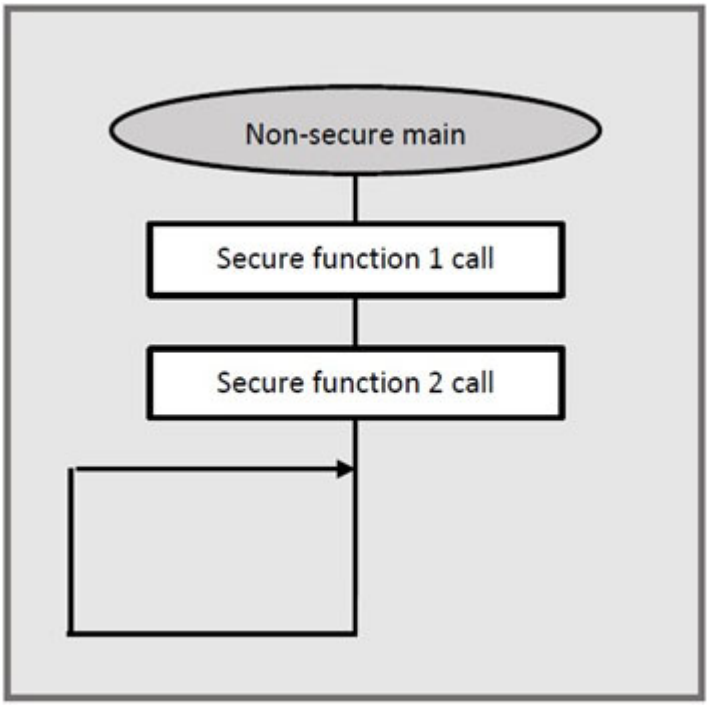
SAM L11 安全解决方案模板中提供的非安全项目为在非安全区域中运行的标准应用程序。该应用程序可使用分配给非安全区域的全部系统资源，具体通过安全应用程序提供的 `veneer.h` 文件来使用预编程非安全可调用（NSC）函数。非安全项目架构如下图所示。

图 2-8. 非安全项目架构



安全解决方案模板中的非安全主函数流程图如下图所示。

图 2-9. 非安全项目主函数流程图



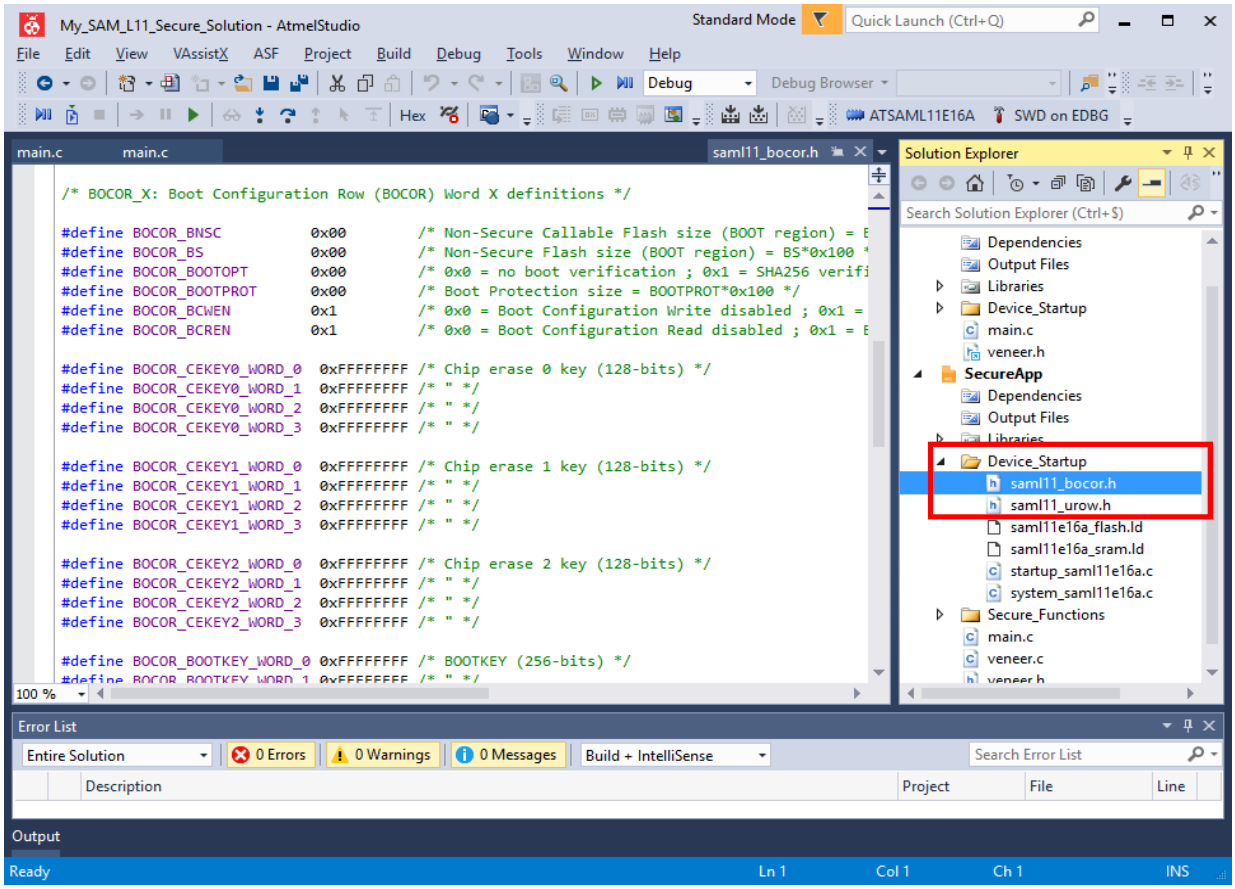
非安全主函数对通过安全应用程序 `veneer.h` 文件提供的网关调用特定安全函数进行了说明。

可以将该非安全 `main.c` 文件用作任何非安全应用程序开发的起始点。

2.3.2.3 NVM 行配置

为了方便定义和修改应用程序熔丝，模板在 SecureApp 项目中嵌入了 2 个专用头文件来管理 SAM L11 系统 NVM 行，如下图所示。

图 2-10. saml11_bocor.h 和 saml11_urow.h



这些熔丝定义引导模式、全片擦除、系统外设（BOD 和看门狗）、IDAU（存储器安全属性）和 PAC（外设安全属性）的配置，必须根据应用需求进行修改。

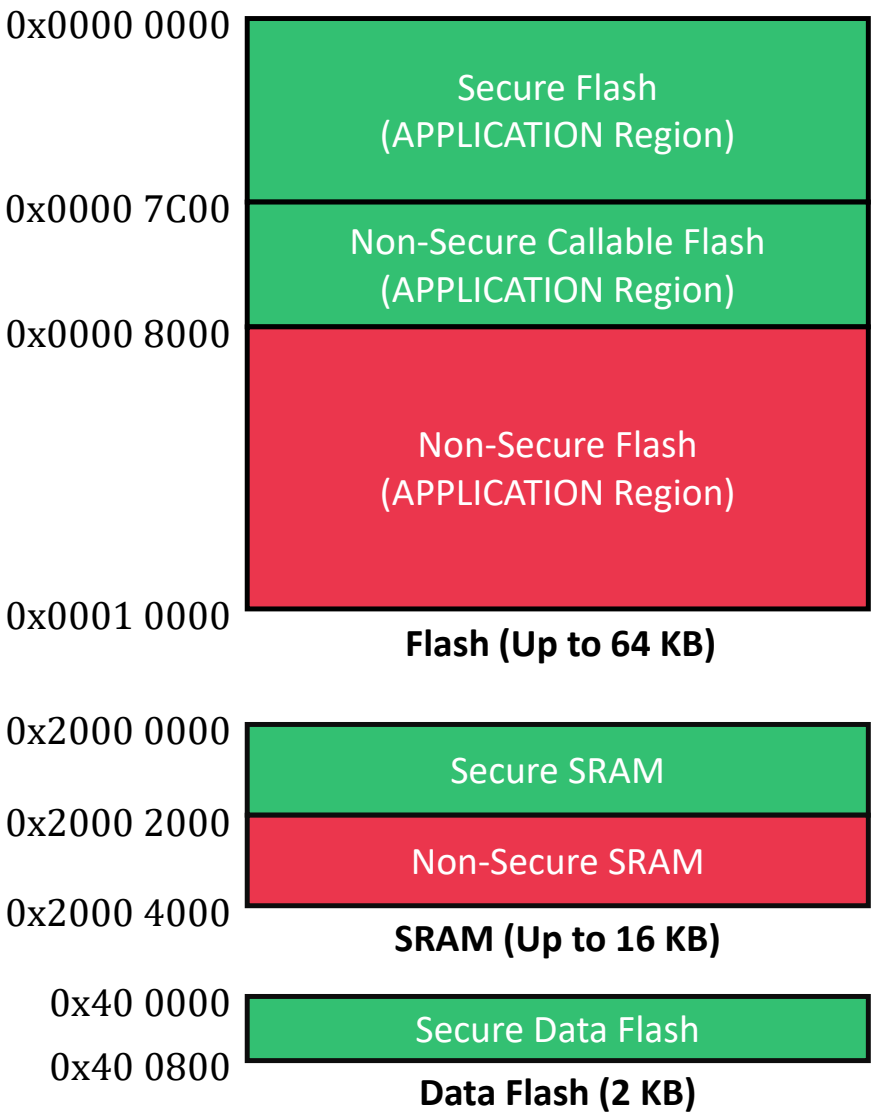
注：有关不同 NVM 行和位域的说明，请参见《SAM L10/L11 数据手册》（DS60001513E_CN）的“NVM 行”一章。

更改熔丝配置需要重启器件，因为熔丝由器件启动时执行的引导 ROM 处理。引导 ROM 负责复制不同外设寄存器中熔丝的配置，然后将配置锁定给任何用户（包括开发人员 A），直到下一次引导为止。

注：有关 SAM L11 引导 ROM 的说明，请参见《SAM L10/L11 数据手册》（DS60001513E_CN）的“引导 ROM”一章。

UROW 和 BOCOR 模板配置与器件默认熔丝配置类似，其相关的存储器映射如下图所示。

图 2-11. SAM L11 安全模板存储器属性

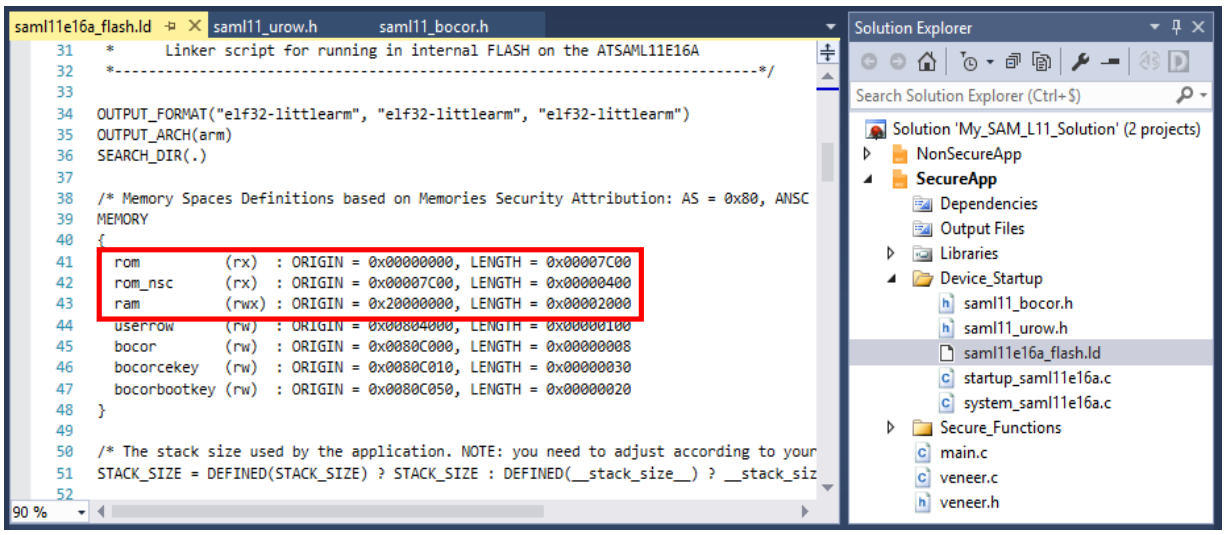


2.3.2.4 安全项目和非安全项目的链接器文件

安全项目和非安全项目都有自己的预配置链接器文件（位于各自的 *Device_Startup* 目录下）。这些文件的内容与 *saml11_urow.h* 和 *saml11_bocor.h* 定义的存储器映射一致，如下图所示。

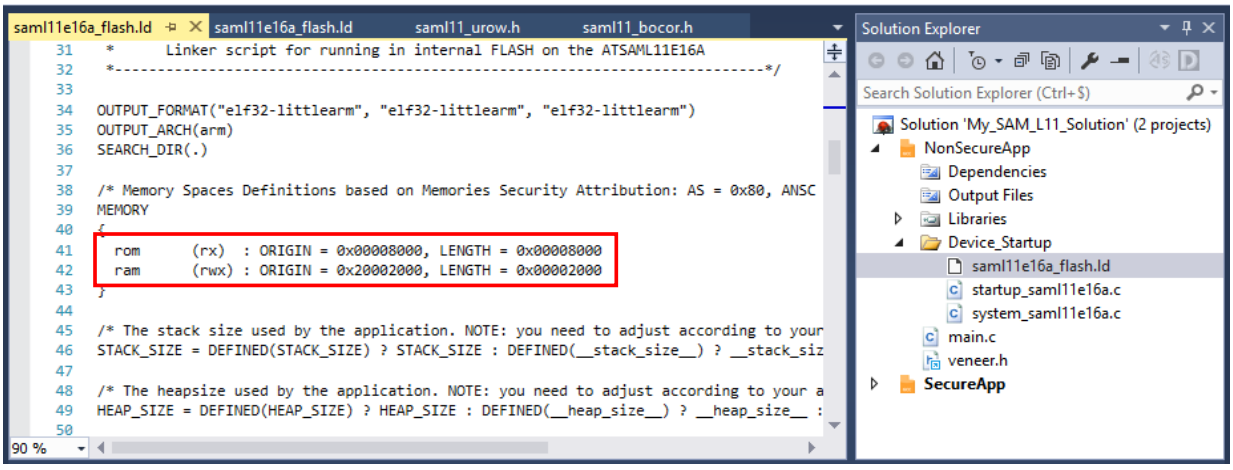
修改熔丝时，务必确保存储器段定义与新的熔丝设置一致，并且非安全存储空间定义与安全存储空间定义之间不存在重叠。下图所示为安全存储空间定义。

图 2-12. 安全存储空间定义



下图所示为非安全存储空间定义。

图 2-13. 非安全存储空间定义



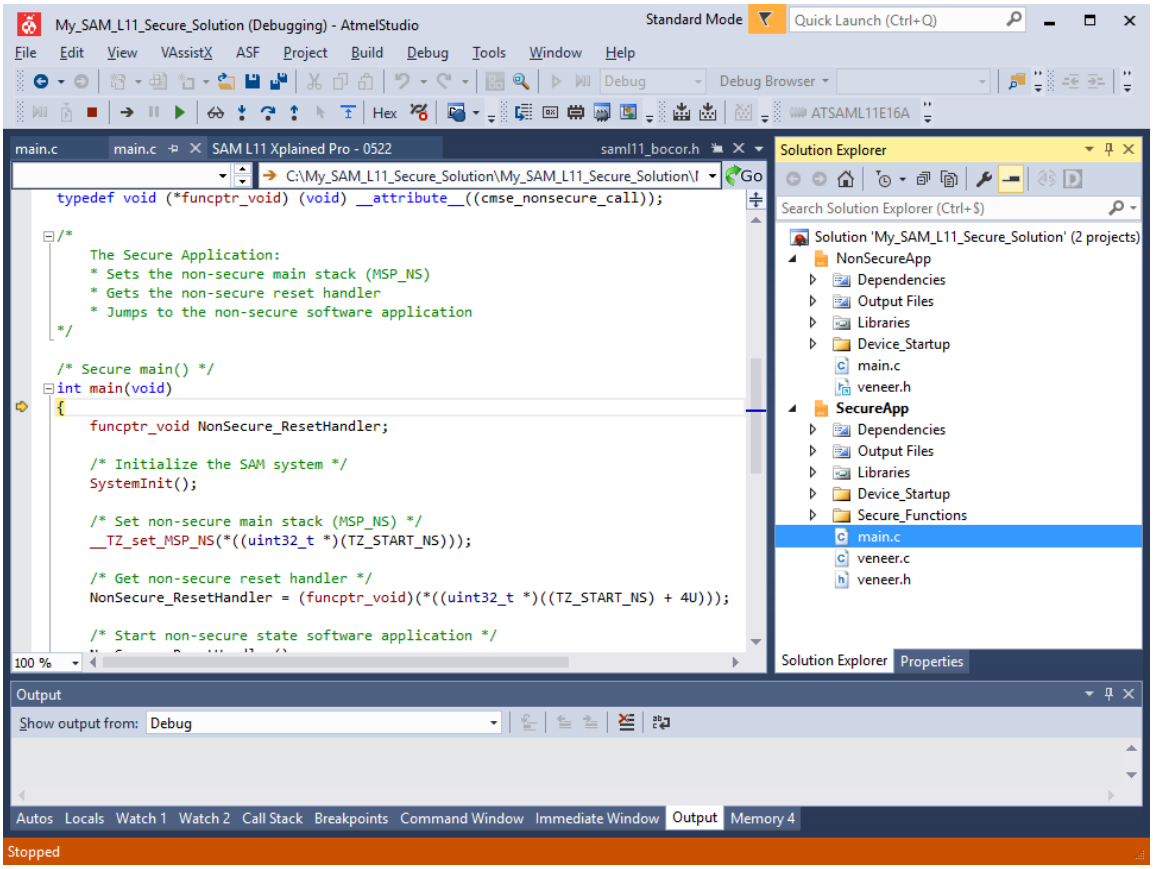
2.3.3 调试安全解决方案

当器件的 DAL = 2 时，允许调试完整解决方案（安全项目 + 非安全项目）。可按照以下步骤使用 Microchip Studio 7 集成开发环境的调试功能来调试 TrustZone 应用程序。

1. 在 Microchip Studio 7 下编译解决方案。
   注：由于解决方案由两个项目组成，因此务必重新编译并装入完整解决方案以确保器件的存储器内容与两个项目的源代码一致。

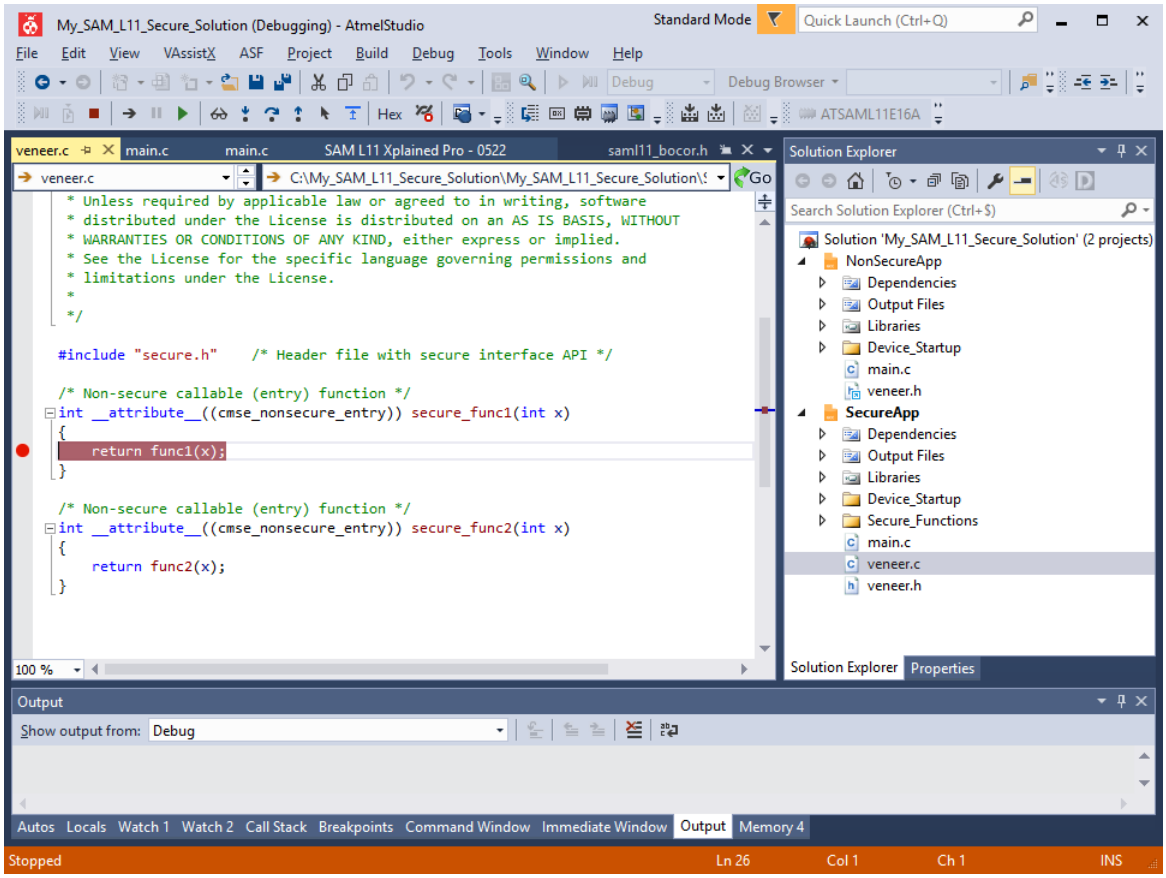
2. 确保调试器连接到计算机和 SAM L11。单击  （Alt + F5）开始调试并在执行安全主函数时自动中断。

图 2-14. 调试并在执行安全主函数时中断



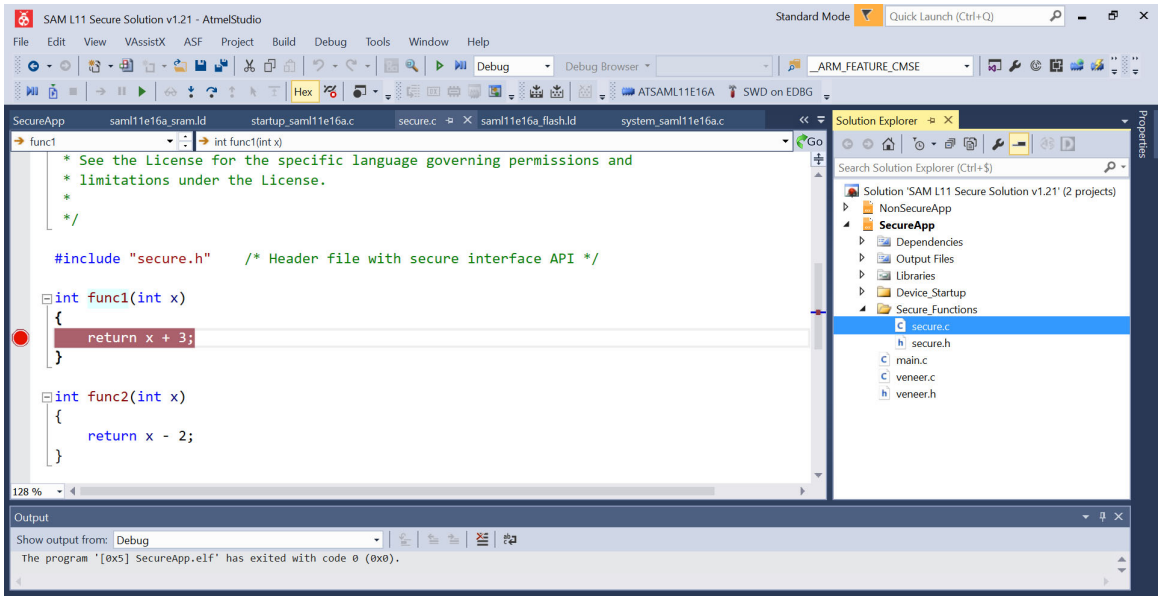
3. 在安全项目 `veneer.c` 文件中的 *secure_func1* 的返回行添加断点。

图 2-15. secure_func1 返回行上的断点（安全项目）



4. 在安全项目 Secure_Functions/secure.c 文件中的 *func1* 的返回行添加断点。

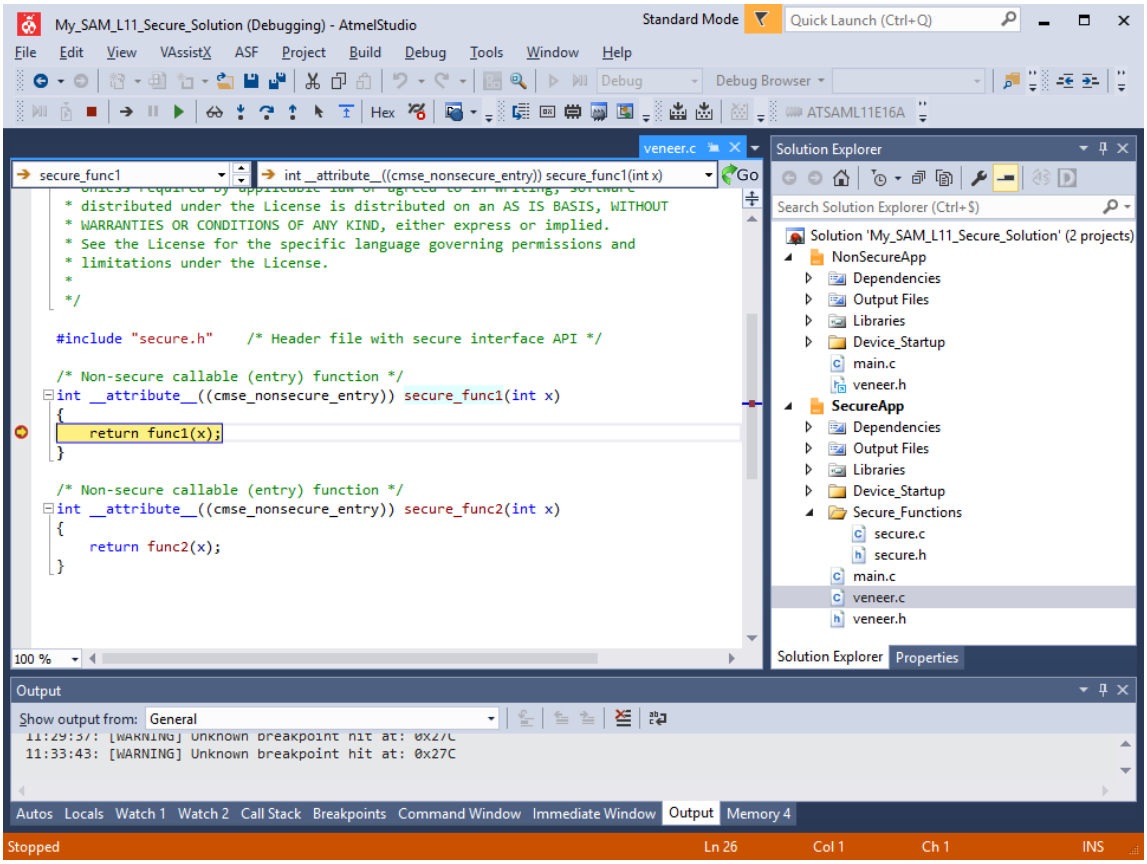
图 2-16. func1 返回行上的断点（安全项目）



⚠ CAUTION 当调试安全应用程序模板时，只能使用硬件断点在执行安全网关（SG）指令时停止执行代码。使用软件断点意味着在 SG 指令前添加断点（BKP）指令，这将在代码执行期间触发安全故障。为确保正常触发该行为，访问 NSC 区域时执行的第一条指令必须为 SG。

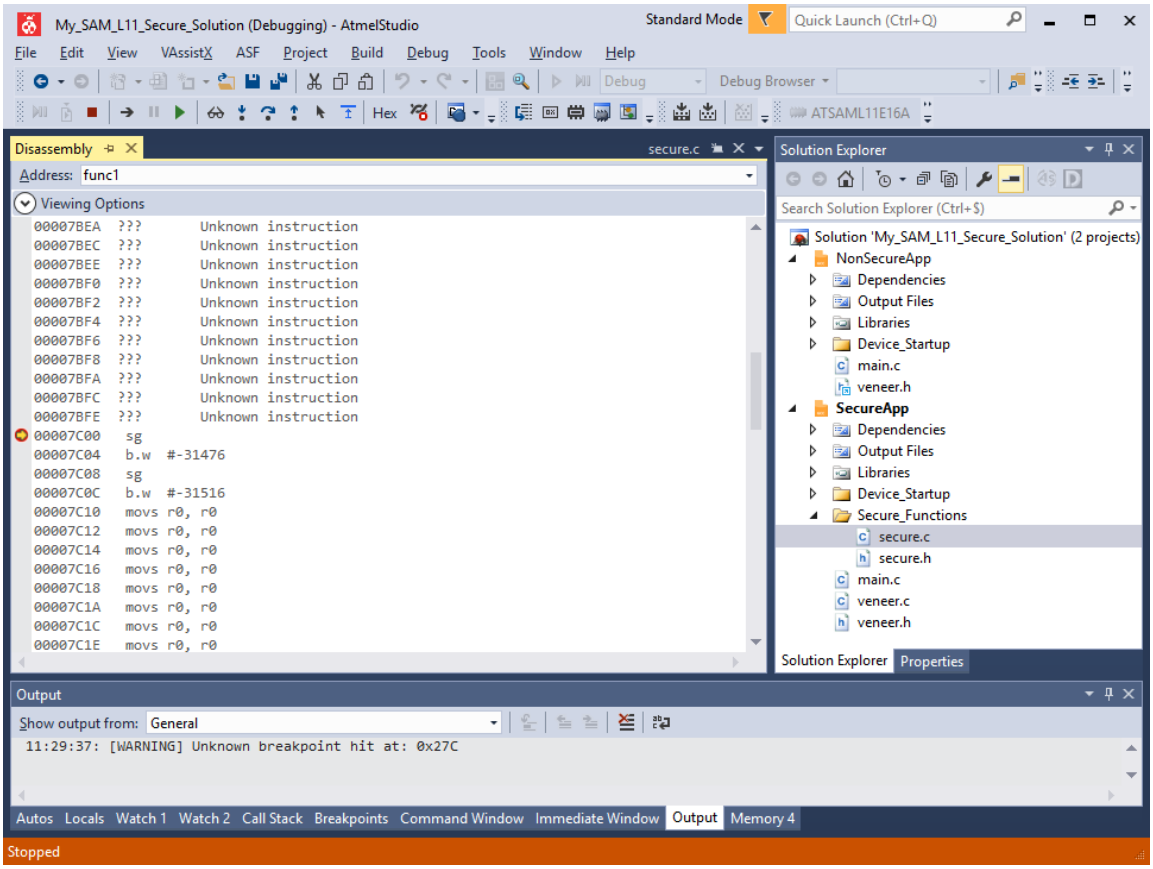
5. 单击 或按下<F5>，继续调试。
最终，调试器一定会在执行以下函数时接连停止：
   - 安全函数模板（安全项目）
   - 安全函数（安全项目）

图 2-17. 在 *secure_func1* 返回行中断



注：可通过选择 Debug > Windows > disassembly（调试 > 窗口 > 反汇编）或按下<Alt + 8>调出包含分步调试功能的代码反汇编窗口。

图 2-18. Microchip Studio 7 反汇编窗口



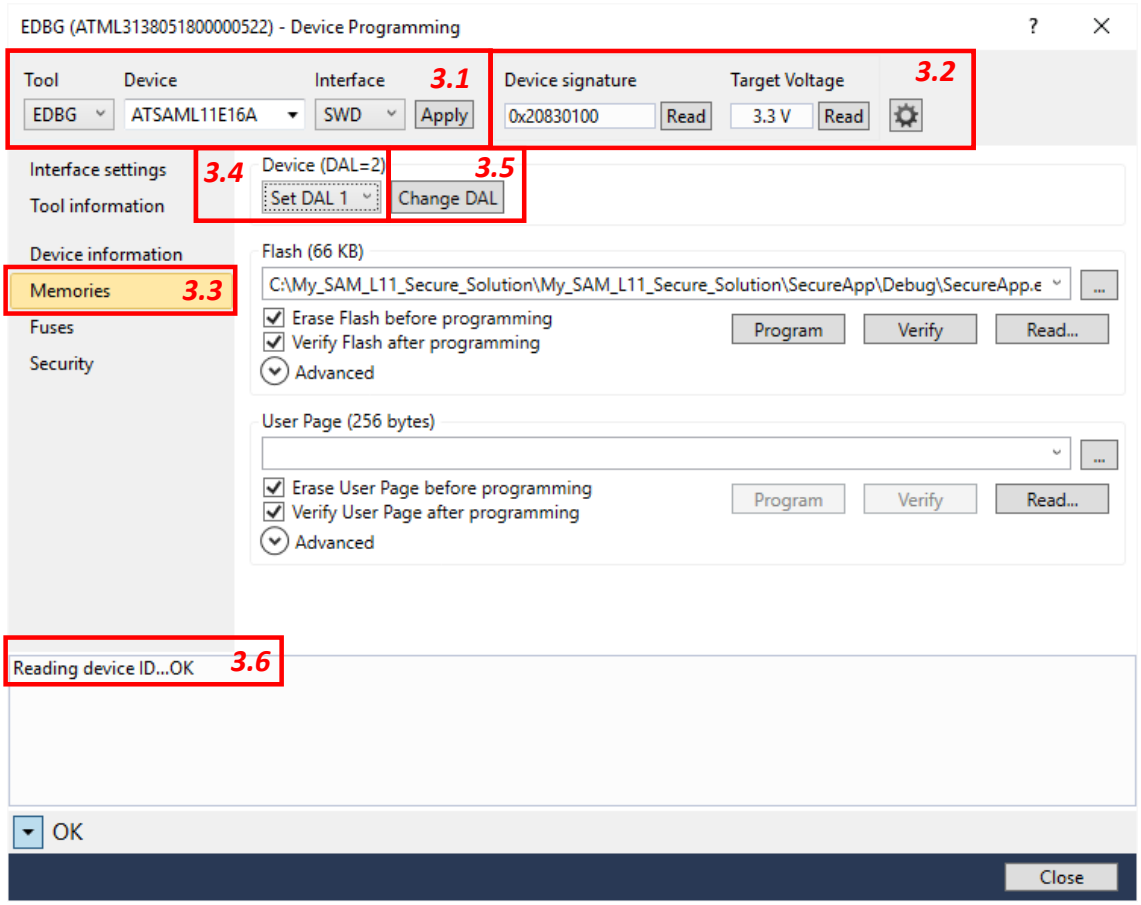
2.3.4 使用调试访问级别保护安全项目

在双开发人员部署方法中，务必确保在将预编程器件提供给开发人员 B 之前防止安全存储区（安全应用程序）遭到进一步的调试器访问。

这可以通过将调试访问级别（DAL）更改为 DAL1 来实现。更改调试访问级别时可以使用器件编程工具，具体步骤如下：

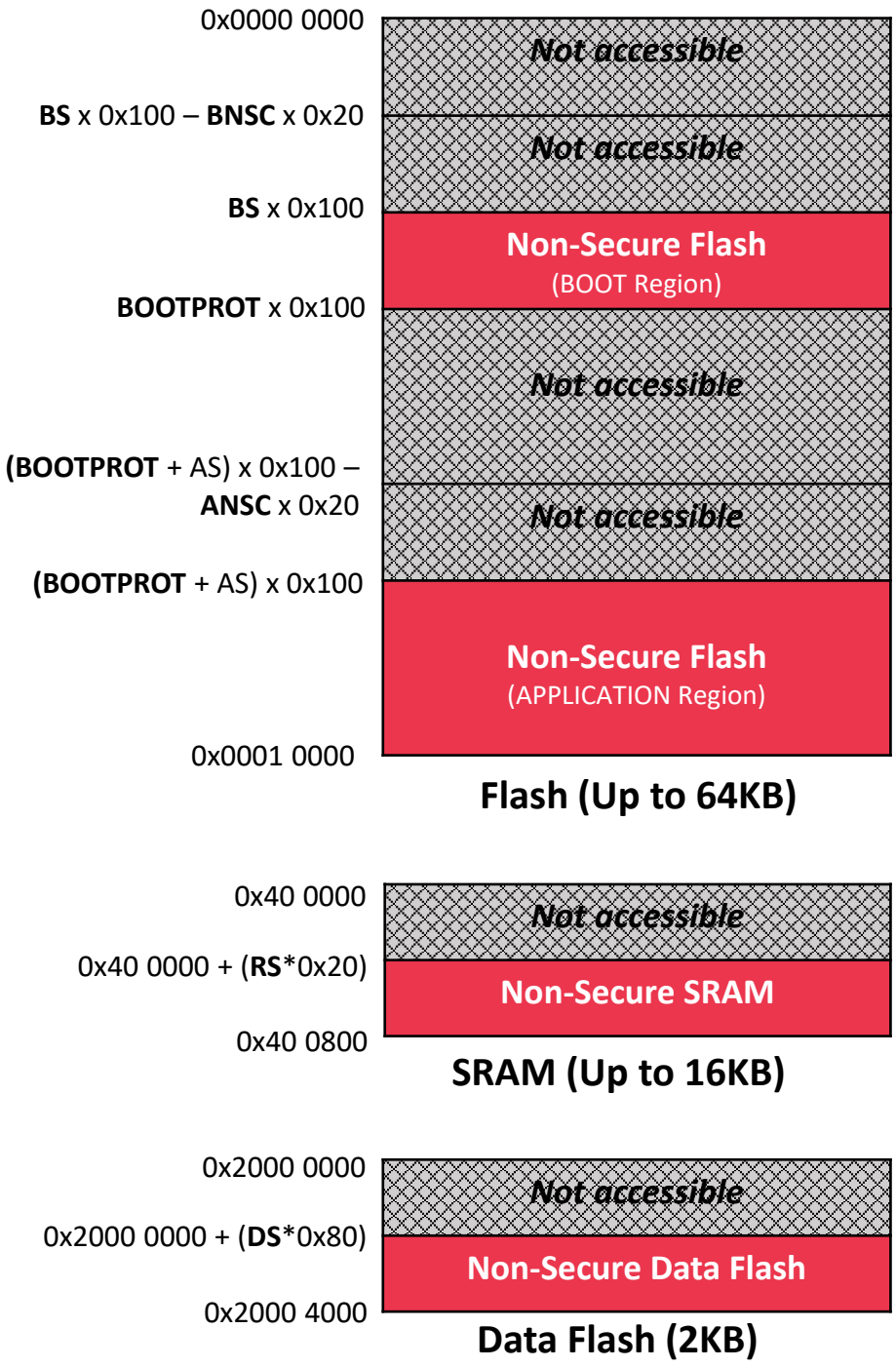
1. 关闭调试会话（如果运行）。
2. 选择 *Tools* > *Device Programming*（工具 > 器件编程），打开器件编程工具。
3. 将 DAL1 命令发送到目标 SAM L11 器件，如下图所示：
   a. 选择 EDBG 器件编程工具，然后单击 **Apply**（应用）。
   b. 在 Device Signature（器件签名）下，单击 **Read**（读取）。
   c. 选择 Memories（存储器）。
   d. 在 Device（器件）下，选择“Set DAL 1”（设置 DAL 1）。
   e. 单击 **Change DAL**（更改 DAL）。
   f. 验证器件编程工具是否报错。

图 2-19. 使用 Microchip Studio 7 器件编程工具更改 DAL



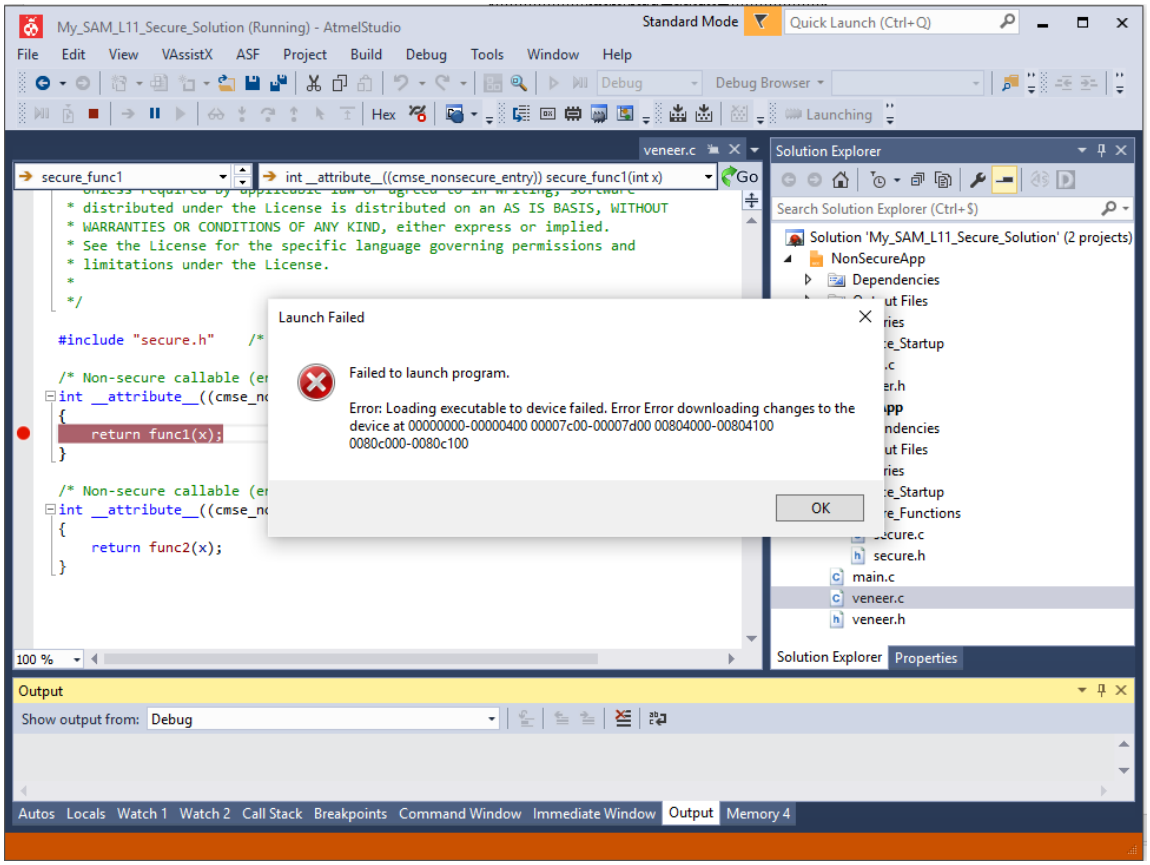
最终，设置 DAL1 可防止后续对器件的安全存储区进行任何调试访问，如下图所示。

图 2-20. DAL 保护器件存储区



后续对安全存储区进行的任何调试访问都将被器件拒绝，同时 Microchip Studio 7 将报错，如下图所示。

图 2-21. DAL 保护区域发生启动失败错误

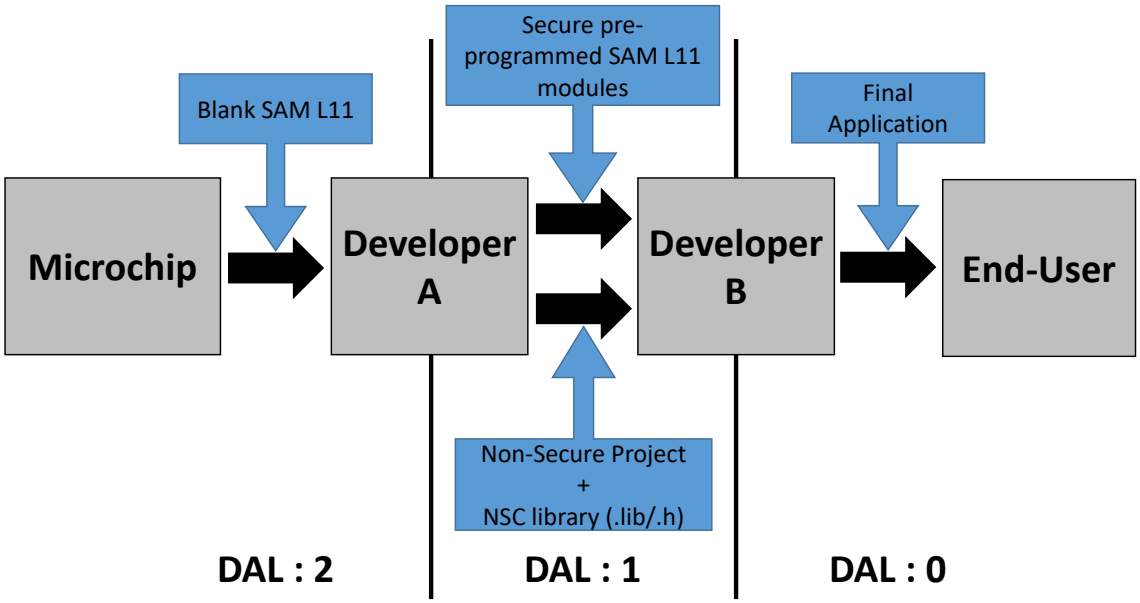


重要：进一步开发器件需要使用独立的非安全项目。请参见创建和配置非安全项目（开发人员 B）。要重新使能对安全存储区的调试访问，必须使用器件编程工具发出 ChipErase_ALL 命令（CE2）。该命令将擦除全部器件存储器和熔丝设置，必须在器件中重新编程安全应用程序。

2.4 开发非安全项目（开发人员 B）

开发人员 B 基于预编程的 SAM L11 器件着手开发，该器件包含具有预定义模板的 DAL1 保护安全项目。更多信息，请参见前一章。

图 2-22. 开发非安全项目（开发人员 B）



在这种情况下，开发人员 A 必须向开发人员 B 提供非安全资源属性说明和非安全可调用函数 API 库。

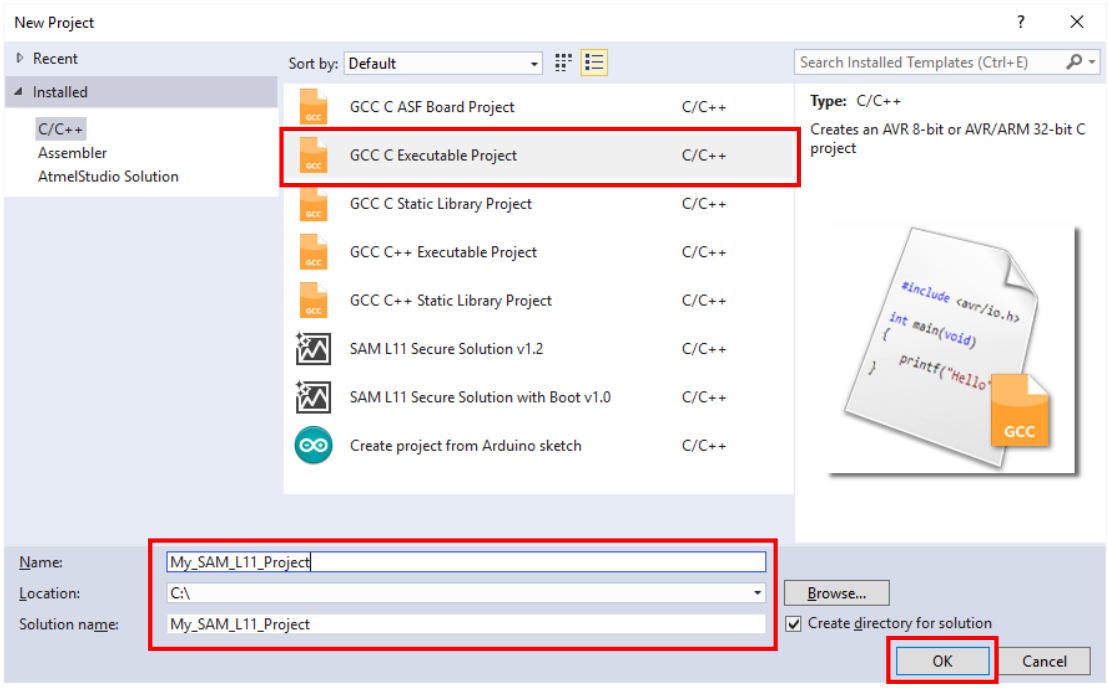
理想情况下，该方法应是由开发人员 A 向开发人员 B 提供非安全项目模板。以下章节介绍了如何为嵌入了预编程 DAL1 保护安全应用程序的 SAM L11 器件创建和配置非安全项目。

2.4.1 创建非安全项目

要使用 Microchip Studio 7 创建非安全项目，请按照以下步骤操作：

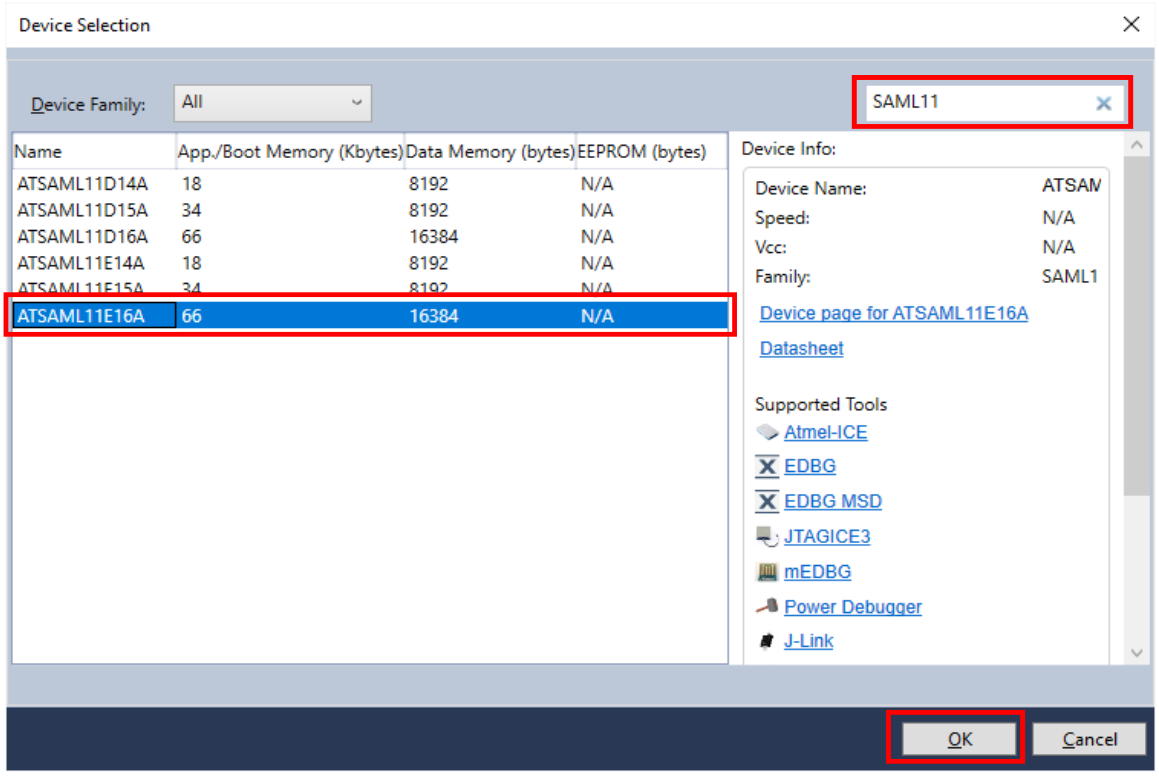
1. 打开 Microchip Studio 7。
2. 选择 *File* > *New* > *Project*。
3. 在 New Project 窗口中，执行以下操作来创建和配置一个新的解决方案：
   a. 展开 Installed，选择 C/C++。
   b. 选择 GCC C Executable Project（GCC C 可执行项目）。
   c. 在 Name、Location 和 Solution name 中输入详细信息（相关示例见下图）。
   d. 单击 **OK**。

图 2-23. 使用 Microchip Studio 7 创建 SAM L11 独立非安全项目



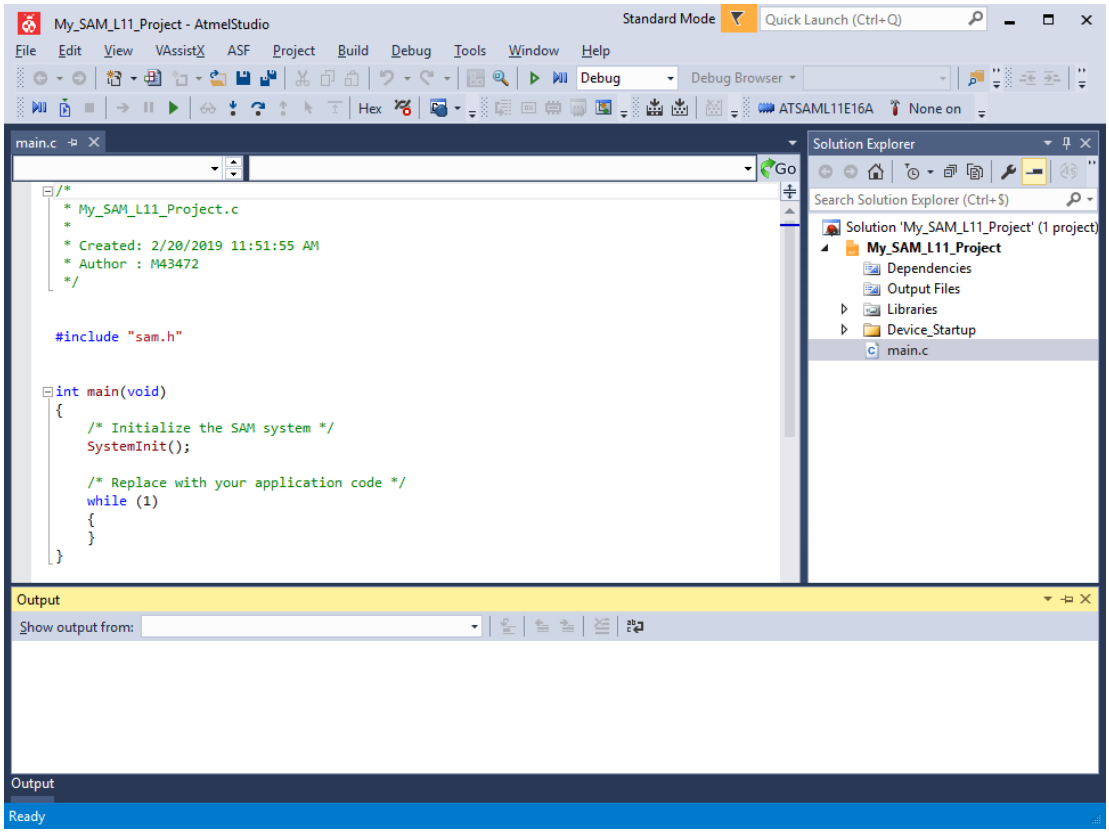
4. 在 Device Selection（器件选型）窗口中选择 ATSAML11E16A 器件，然后单击 **OK**。

图 2-24. 新建 SAM L11 独立非安全项目时的 SAM L11 产品选型



非安全项目将显示在 Microchip Studio 7 IDE 中，如下图所示。

图 2-25. SAM L11 独立非安全项目



2.4.2 项目配置

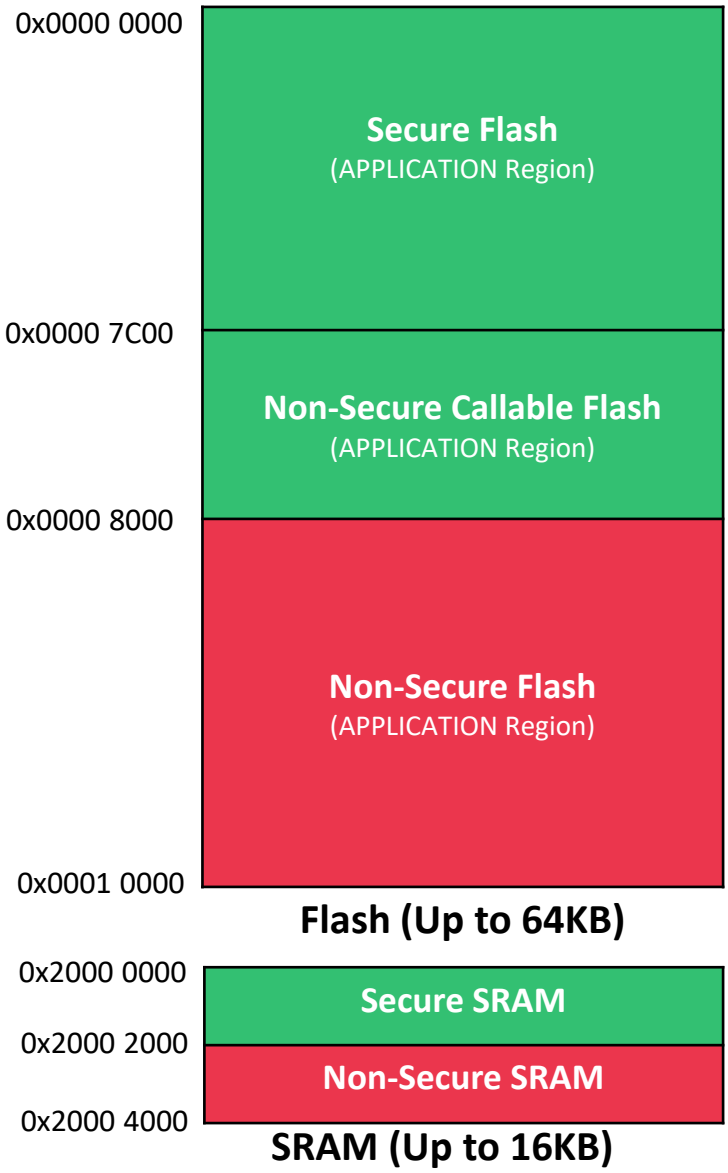
创建非安全项目后，按照以下步骤根据预编程安全项目映射和安全网关 API 对其进行配置：

- 配置项目，使其链接器文件与开发人员 A 预定义的安全和非安全存储器属性一致。
- 将安全网关库链接到项目并将模板头文件添加到项目。

2.4.2.1 使项目链接器文件与 SAM L11 非安全存储器属性一致

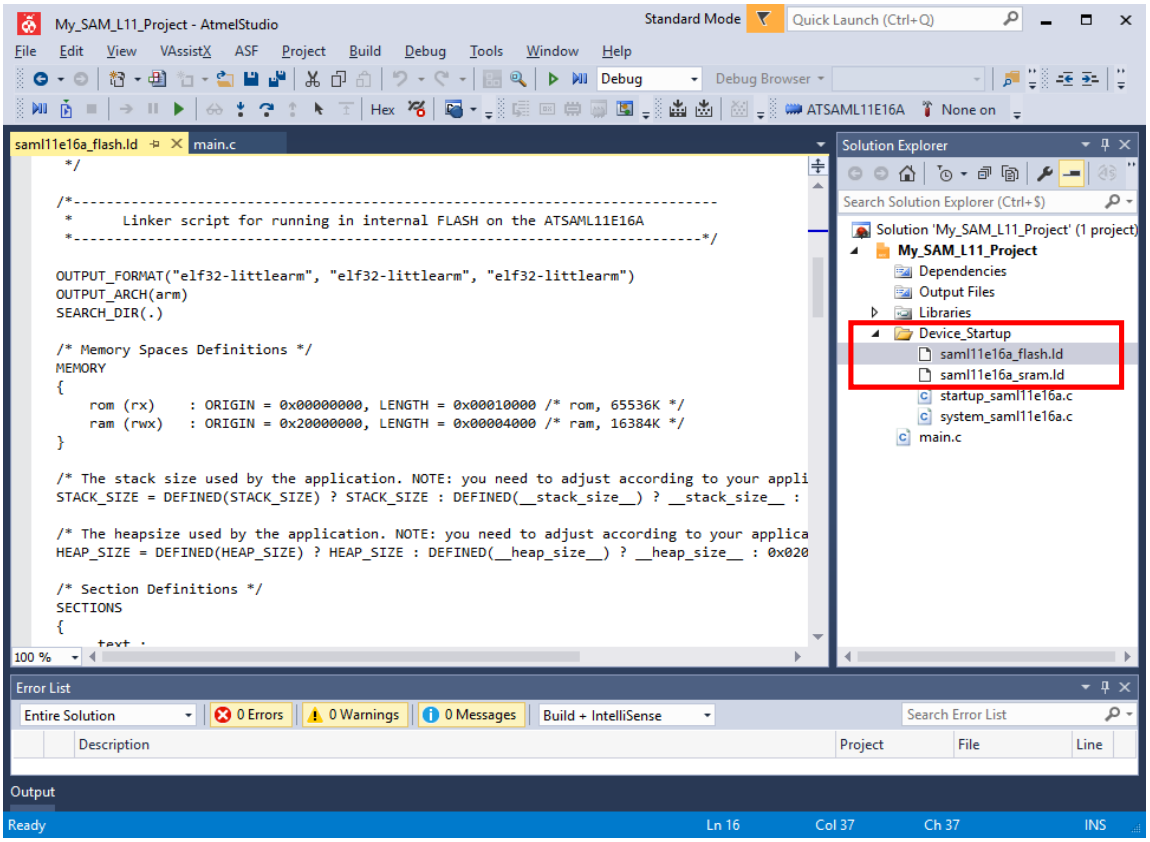
按照以下步骤根据下图所示的安全和非安全存储空间分配修改非安全解决方案项目链接器文件。

图 2-26. 安全和非安全存储空间



1. 打开项目链接器文件：Device Startup/saml11e16a_flash.ld。

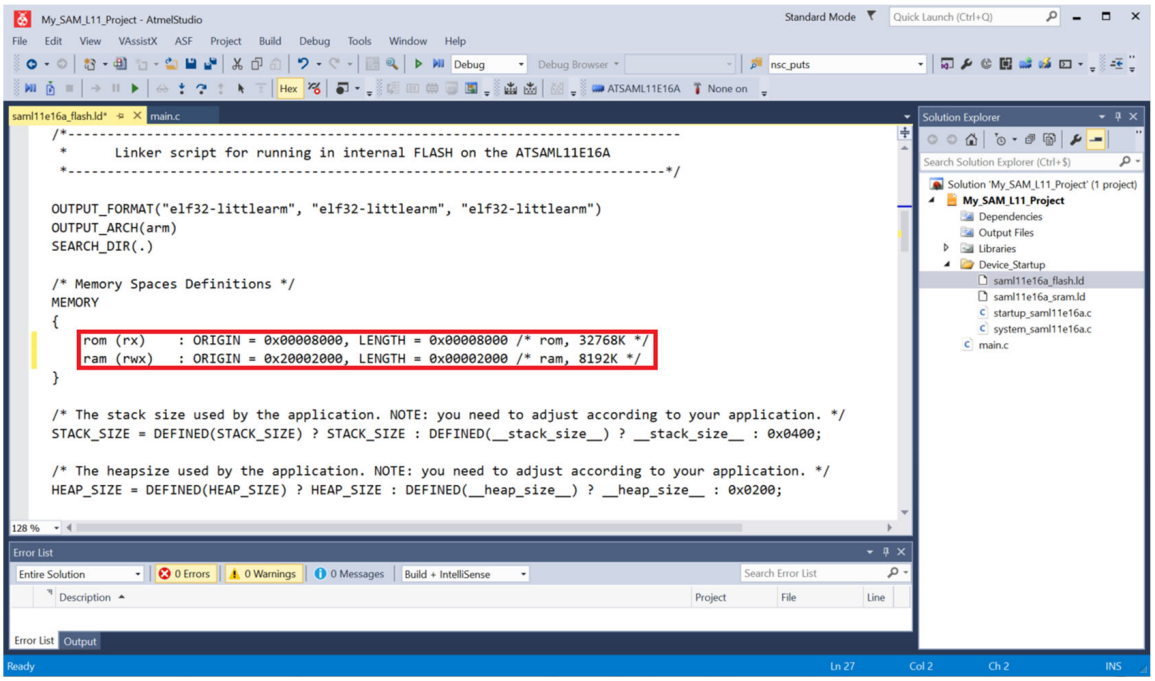
图 2-27. 非安全项目链接器文件位置



2. 根据 SAM L11 非安全存储器属性更新链接器文件存储空间定义。

```
/* 存储空间定义 */
MEMORY
{
rom       (rx)   : ORIGIN = 0x00008000, LENGTH = 0x00008000
ram       (rwx)  : ORIGIN = 0x20002000, LENGTH = 0x00002000
}
```

图 2-28. 非安全存储器地址和大小定义

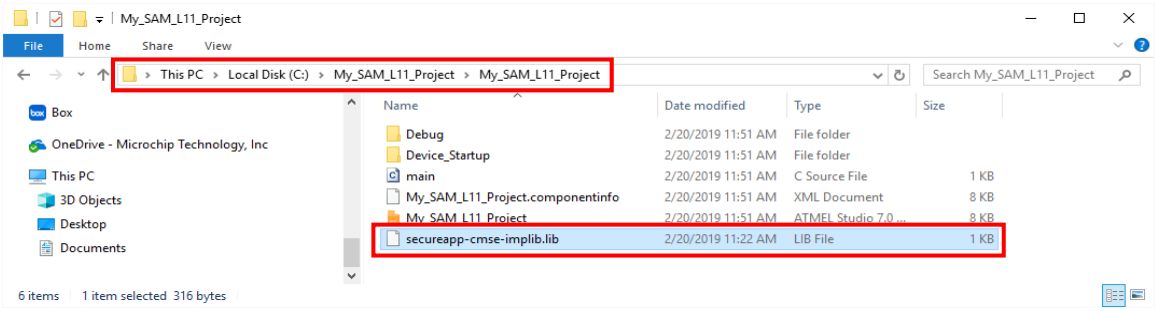


2.4.2.2 将安全网关库添加并链接到非安全项目

按照以下步骤添加并链接开发人员 A 在安全应用程序开发期间生成的安全网关库：

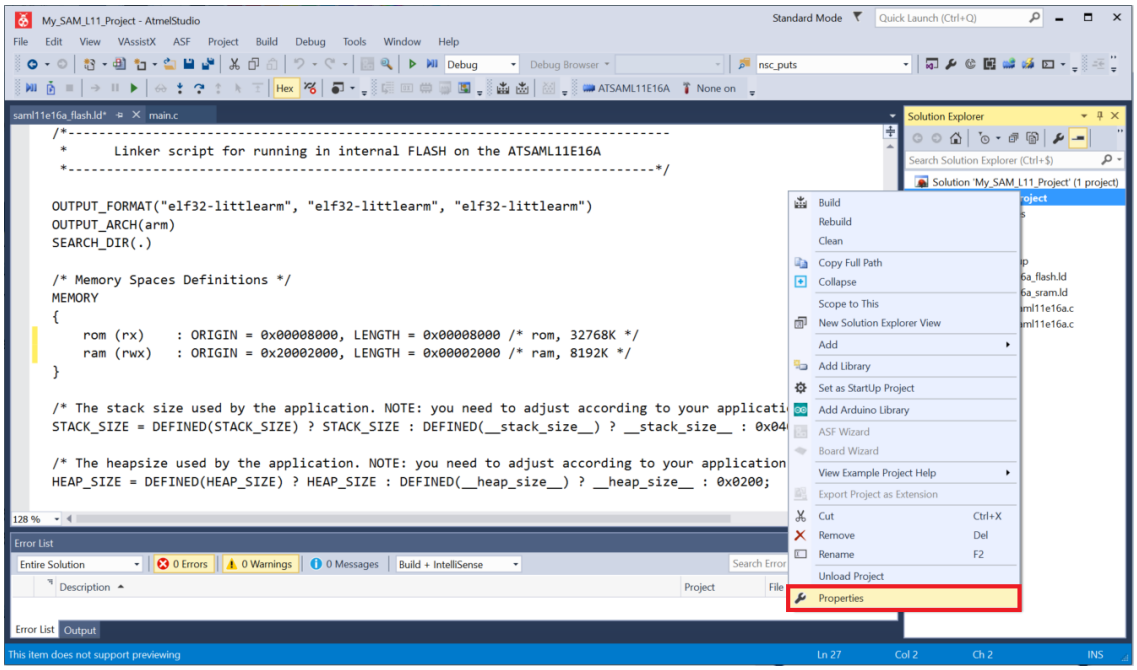
1. 将安全项目 `implib` 复制到非安全项目中。

图 2-29. 将安全网关库文件添加到非安全项目源中



2. 在 Microchip Studio 7 中，右键单击非安全项目，然后选择 Properties（属性）。

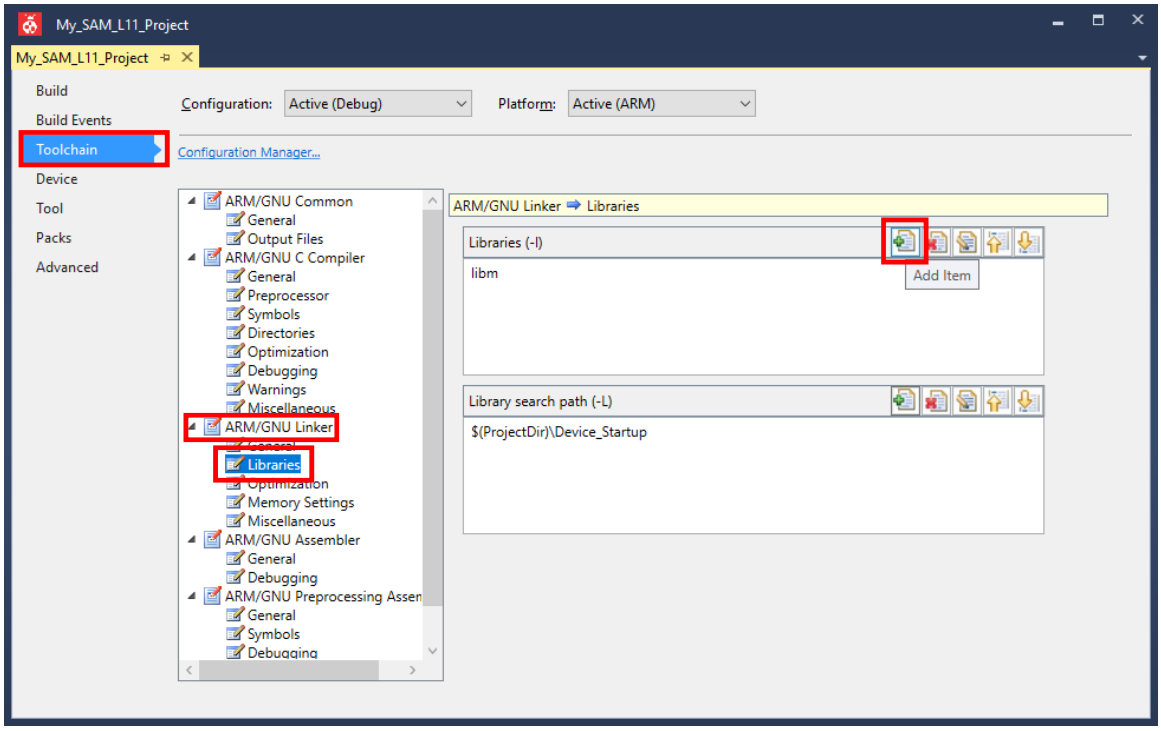
图 2-30. 访问非安全项目属性



3. 要添加安全项目库，请选择 ToolChain（工具链）并展开 ARM/GNU Linker（ARM/GNU 链接器），然后选择 Libraries（库）。

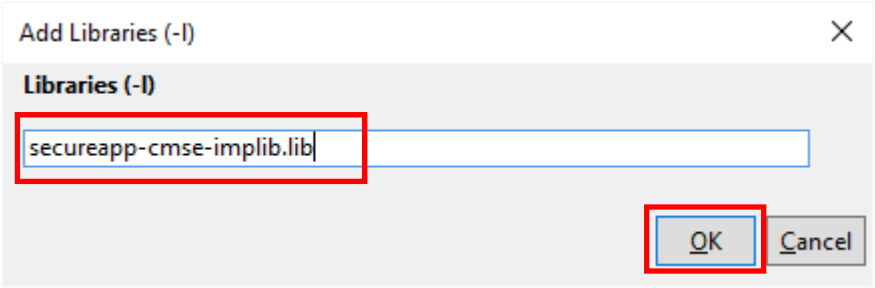
4. 单击  （Add Item（添加项）按钮）。

图 2-31. 将新库添加到链接选项中



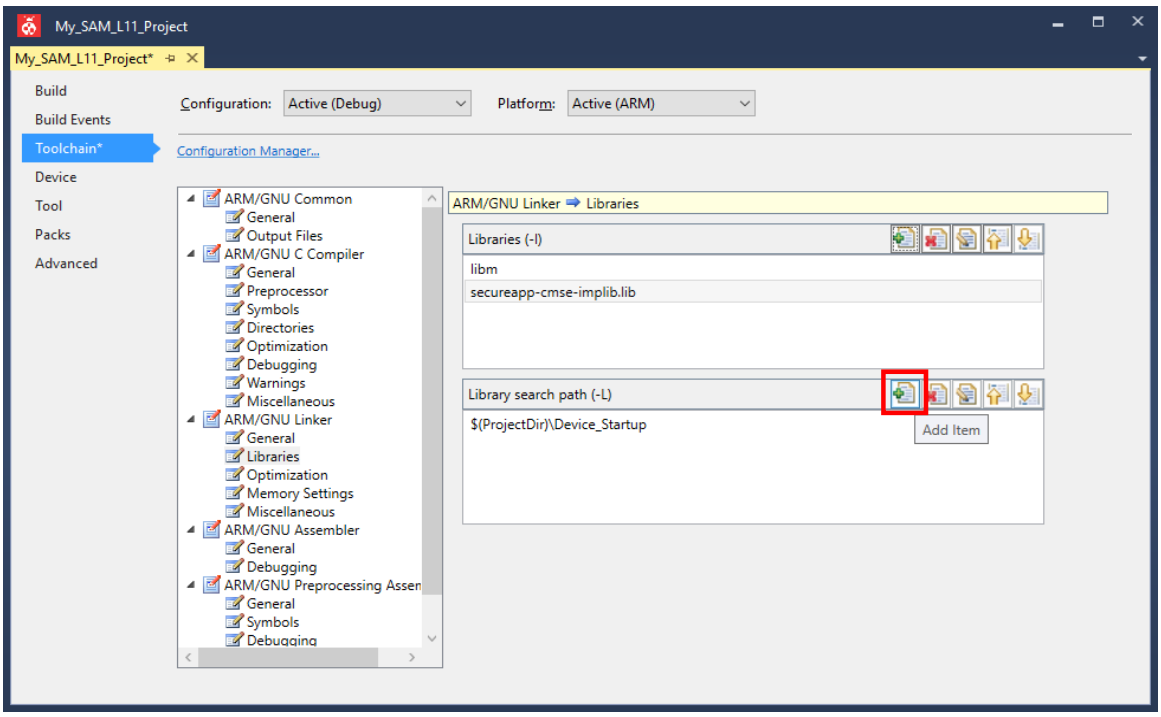
5. 在 Add Libraries（添加库）对话框中，按如下所示输入库名称，然后单击 **OK**。

图 2-32. 添加安全网关库名称



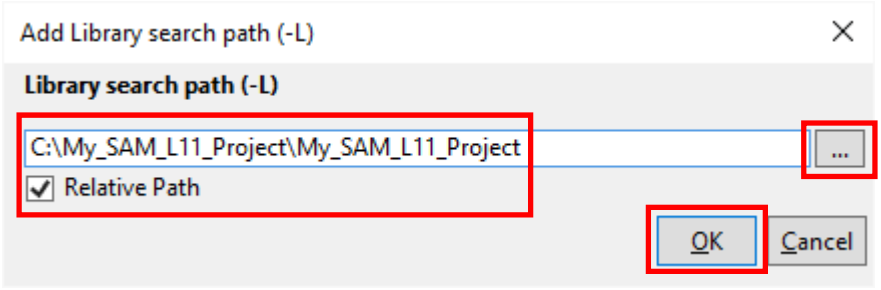
6. 要添加安全项目库路径，请选择 *Toolchain* > *ARM/GNU Linker* > *Libraries*（工具链 > ARM/GNU 链接器 > 库）。
7. 单击  （Add Item 按钮）。

图 2-33. 添加新的库搜索路径



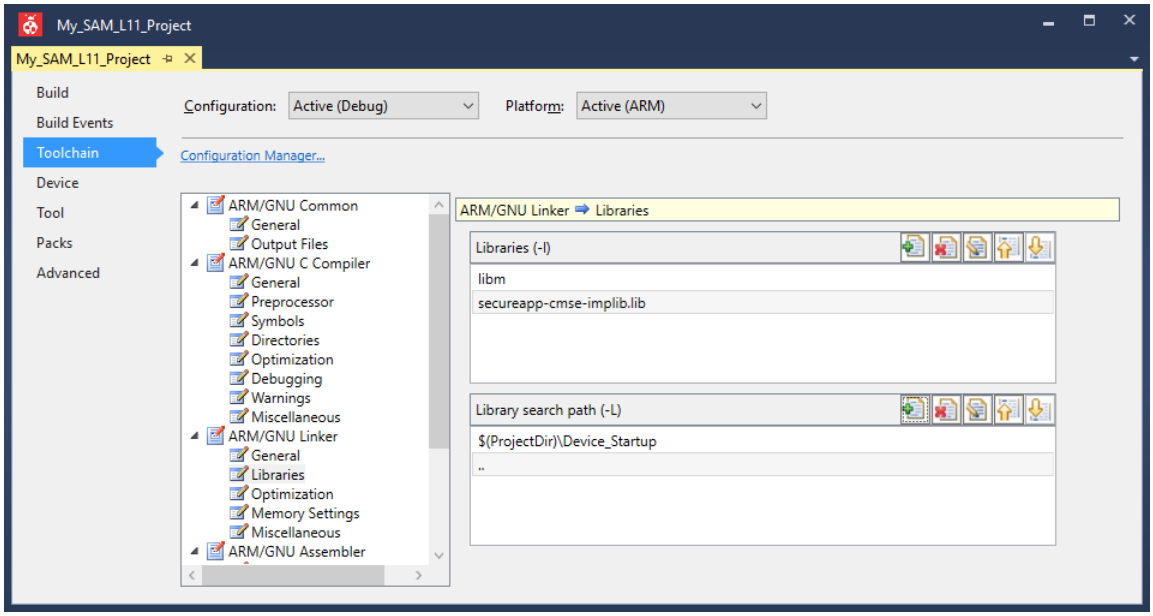
8. 在 Add Library search path（添加库搜索路径）对话框中，选择安全项目 implib 的位置。
9. 选择 Relative Path（相对路径）以确保项目可移植性。
10. 单击 **OK**。

图 2-34. 输入安全网关库的相对路径



11. 链接器库属性将如下图所示：

图 2-35. 非安全项目链接器库配置



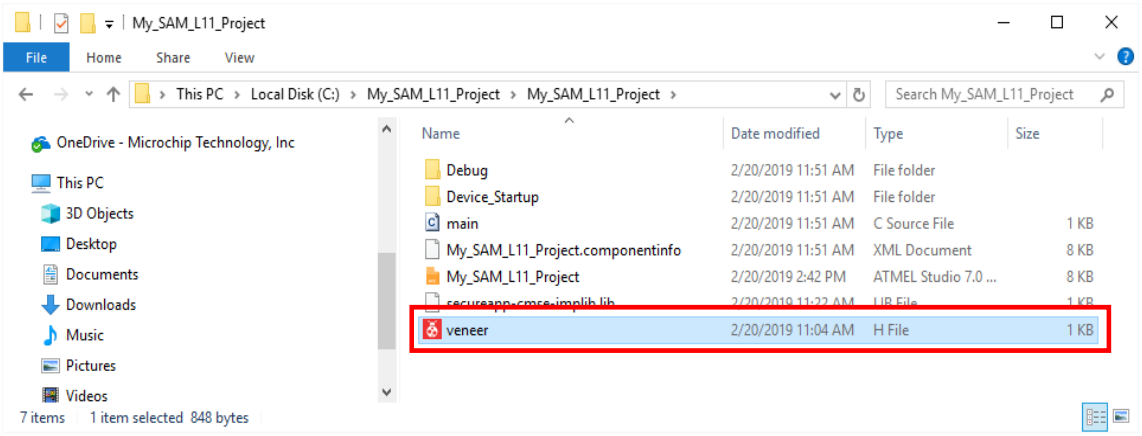
12. 单击  （Save（保存）按钮）保存项目设置。

2.4.2.3 添加并包含安全网关头文件

要添加并包含安全网关头文件，请按照以下步骤操作：

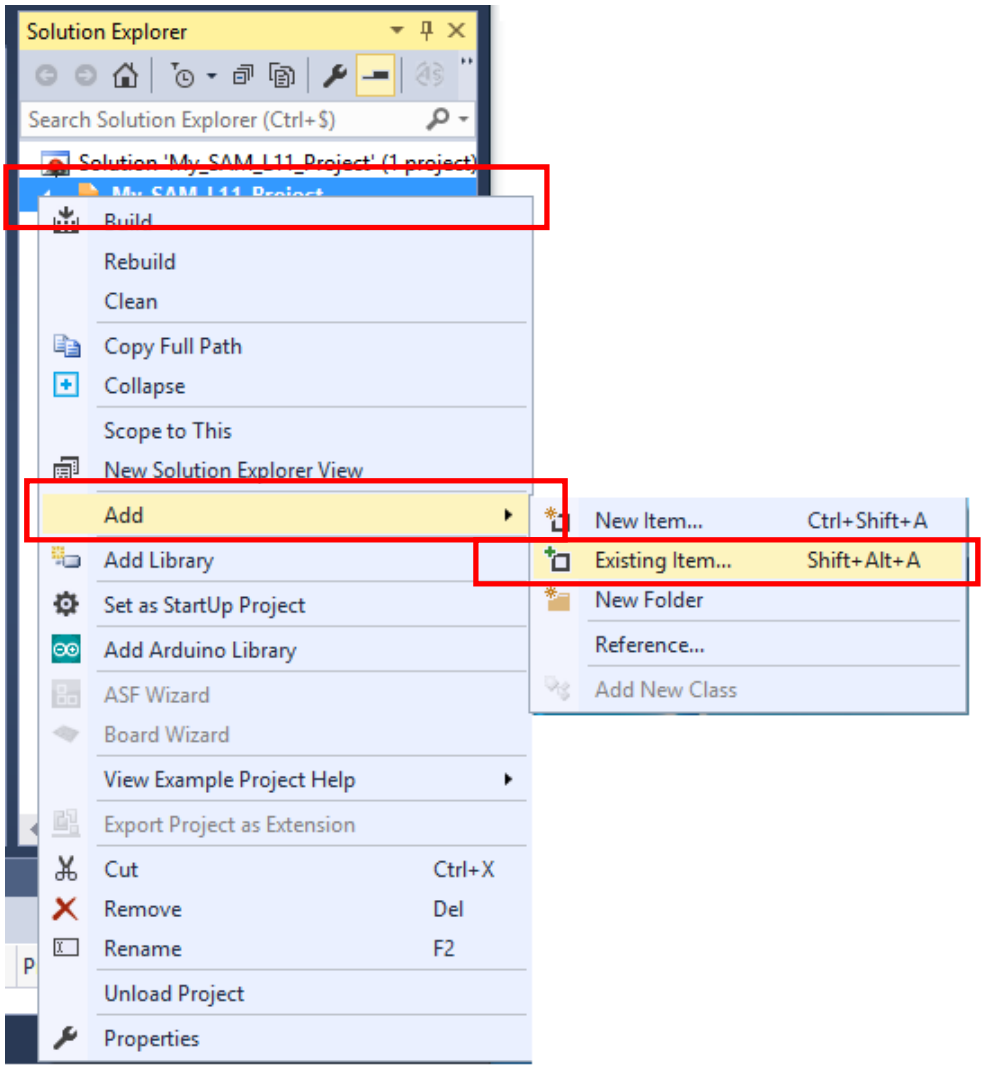
1. 将安全项目中的安全网关头文件复制到非安全项目中。

图 2-36. 将安全网关头文件包含在非安全项目源中



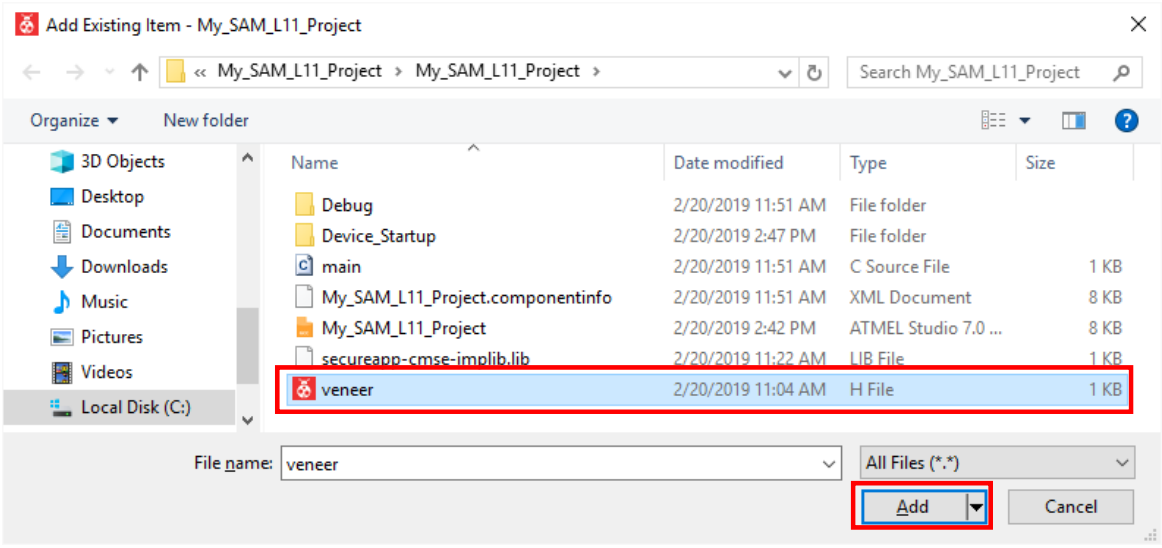
2. 右键单击 Solution Explorer（解决方案资源管理器）中的**非安全项目**，然后选择 *Add* > *Existing Item*（添加 > 现有项）。

图 2-37. 将安全网关头文件包含在 Microchip Studio 7 Solution Explorer 中



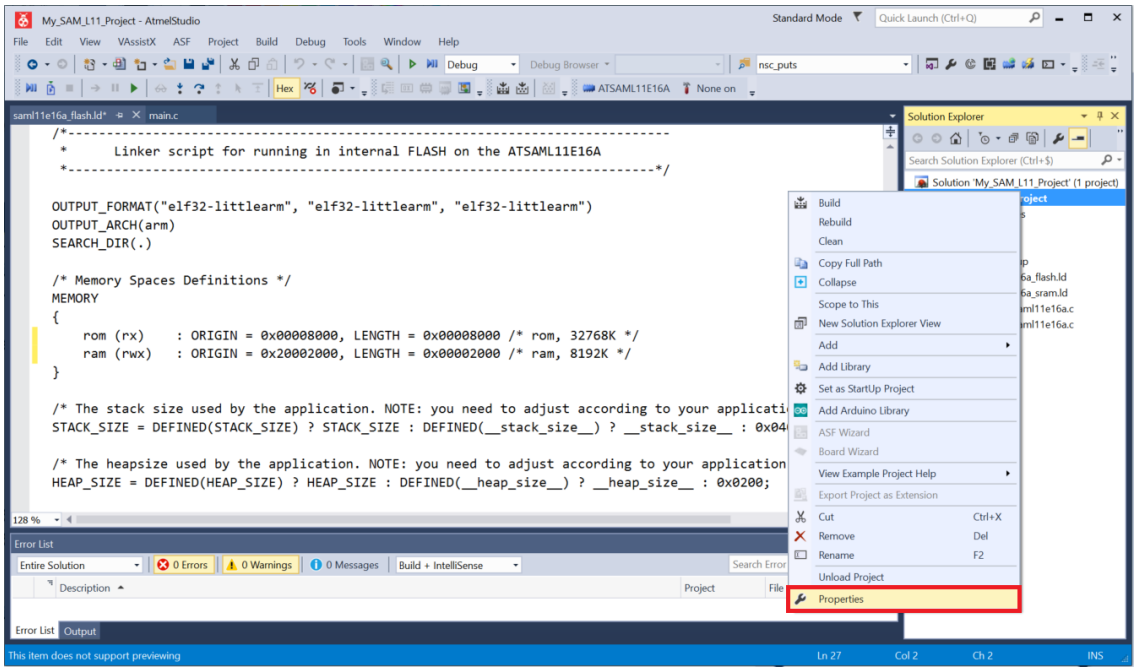
3. 选择安全网关头文件，然后单击 **Add**（添加）。

图 2-38. 将安全网关头文件包含在非安全项目中



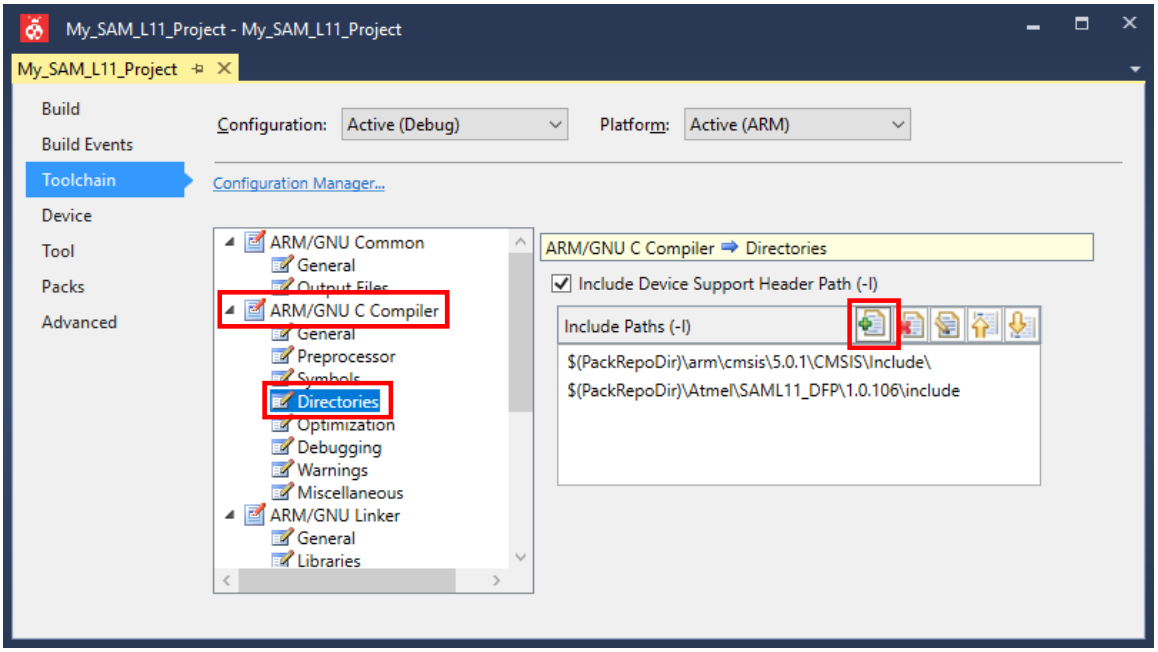
4. 右键单击 Solution Explorer 中的**非安全项目**，然后选择 **Properties**。

图 2-39. 在 Microchip Studio 7 下访问非安全项目属性



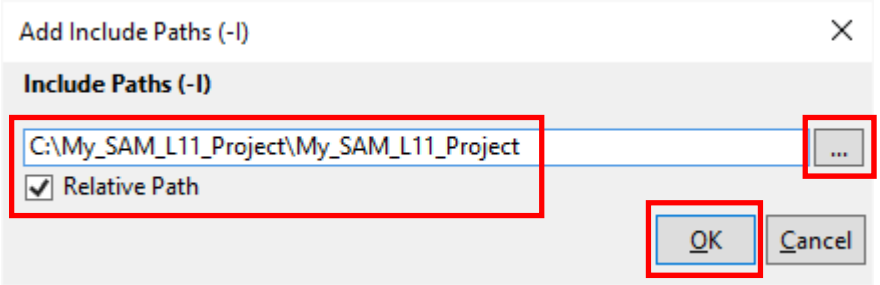
5. 在非安全项目窗口中，选择 *Toolchain* > *ARM/GNU C Compiler* > *Directories*（工具链 > ARM/GNU C 编译器 > 目录），然后单击  （Add Item 按钮）。

图 2-40. 将新的编译器目录添加到非安全项目中



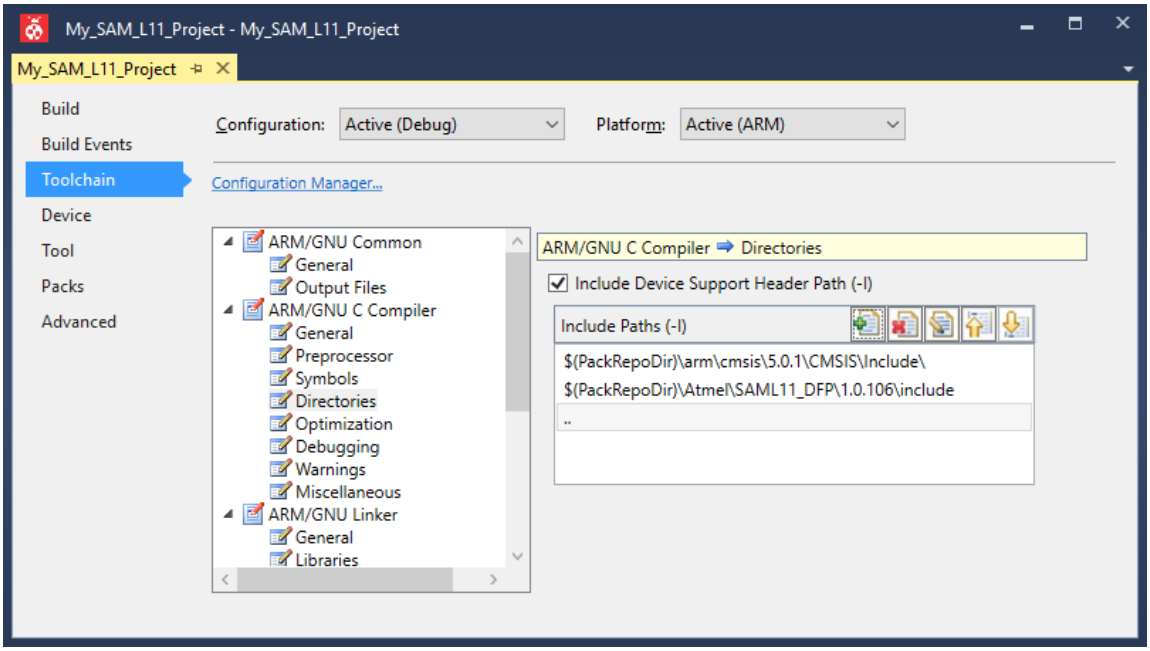
6. 在 Add Include Paths（添加包含路径）对话框中，选择 `veneer.h` 文件的位置。
7. 选择 Relative Path 以确保项目可移植性，然后单击 **OK**。

图 2-41. 将安全网关库路径包含在编译器目录中



8. 将显示非安全项目编译器目录属性。

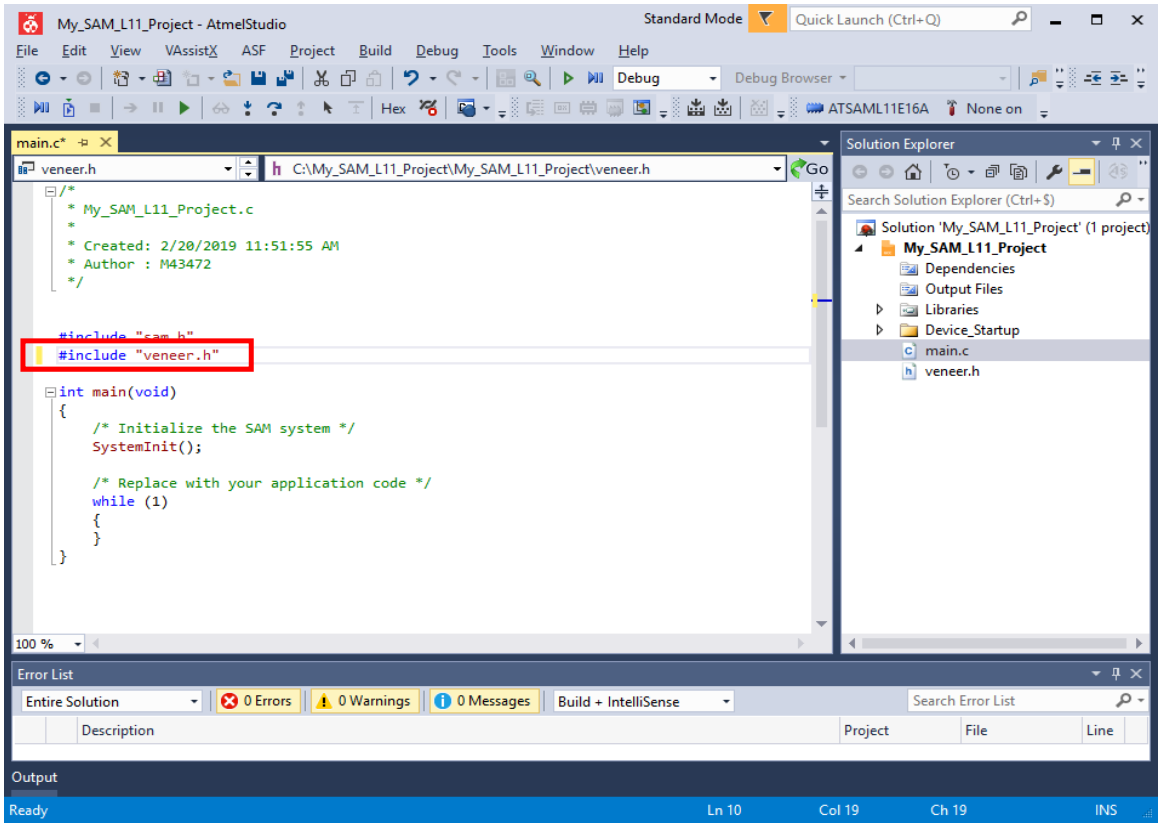
图 2-42. 非安全项目编译器目录参数



9. 按下  （Save 按钮）保存项目设置。

10. 要添加安全网关库，请将突出显示的代码添加到 main.c 文件的开头。

图 2-43. 将 veneer.h 包含在非安全项目 main.c 文件中



11. 单击  （Save 按钮）将修改保存到 main.c 文件中。
12. 单击  （Build Project（编译项目）按钮）。
13. 验证编译过程是否未报错。

重要：在将项目装入目标 SAM L11 器件之前，务必选中 *Project Properties > Tools > Programing settings*（项目属性 > 工具 > 编程设置），确保编程过程不会在装入应用程序前执行 ChipErase_All 命令。理想配置为“Erase only Program area”（仅擦除编程区域），如下图所示。

图 2-44. 项目编程设置

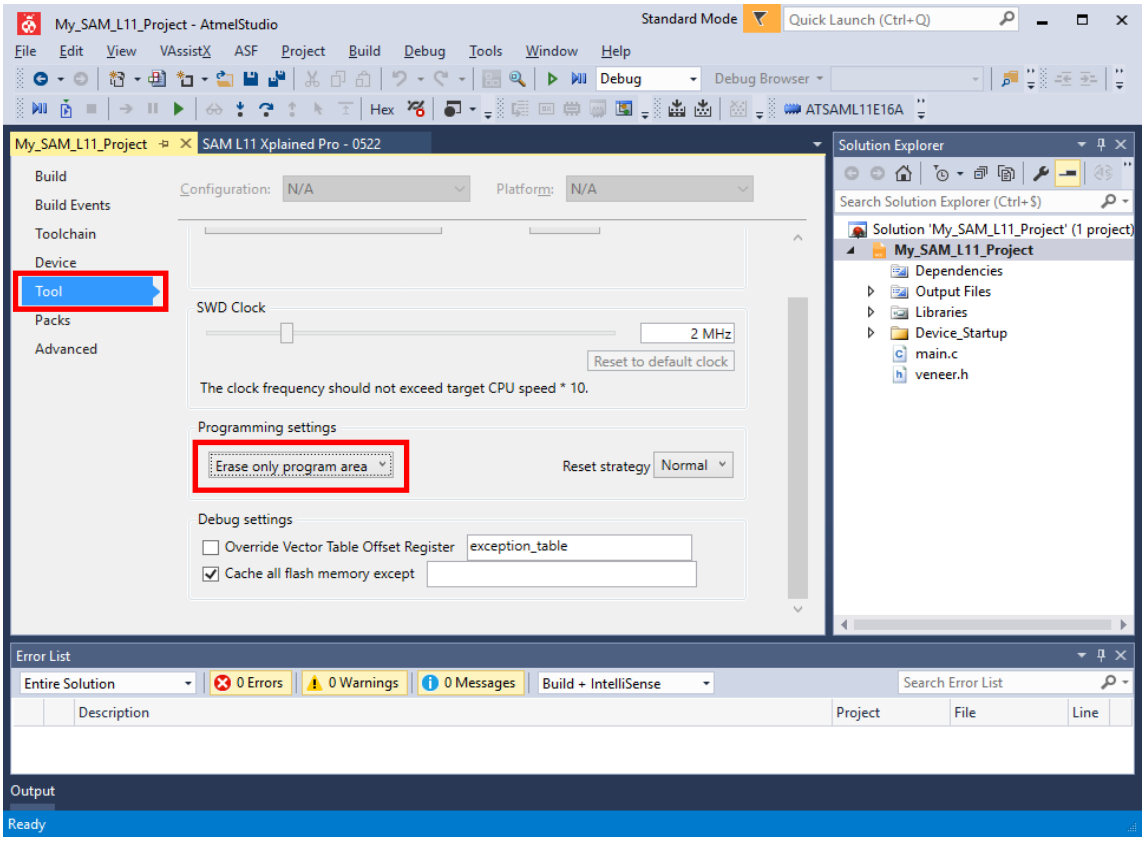
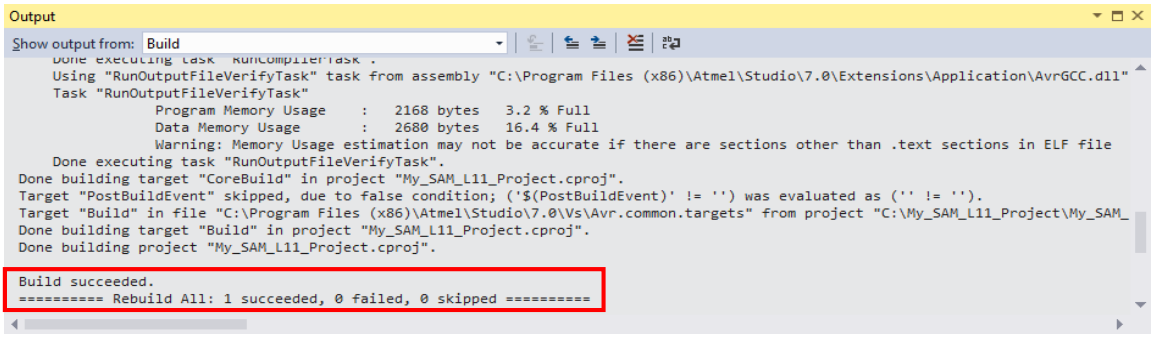


图 2-45. 非安全项目成功编译

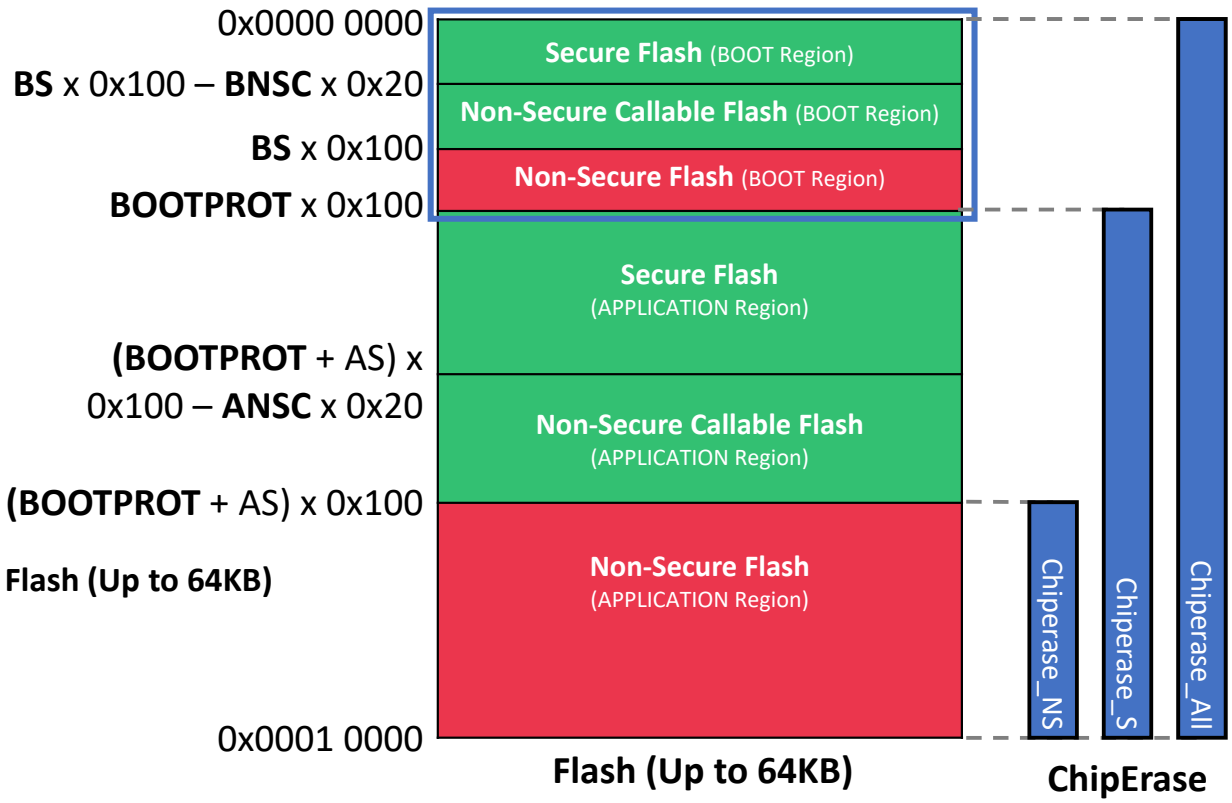


14. 启动调试会话并验证项目是否正在工作。

重要：调试非安全项目需要使用兼容的预编程安全应用程序，该应用程序用于配置并启动非安全执行。如果 MCU 上没有该安全应用程序，则调试过程将挂起。

2.5 开发具有安全引导程序的解决方案（开发人员 A）

SAM L11 器件提供了两个可配置的存储器段来分别存储安全引导程序和非安全引导程序。这两个存储器段均具有 ChipErase_S 和 ChipErase_NS 保护，具体可按下图所示存储安全自举程序代码和非安全自举程序代码。

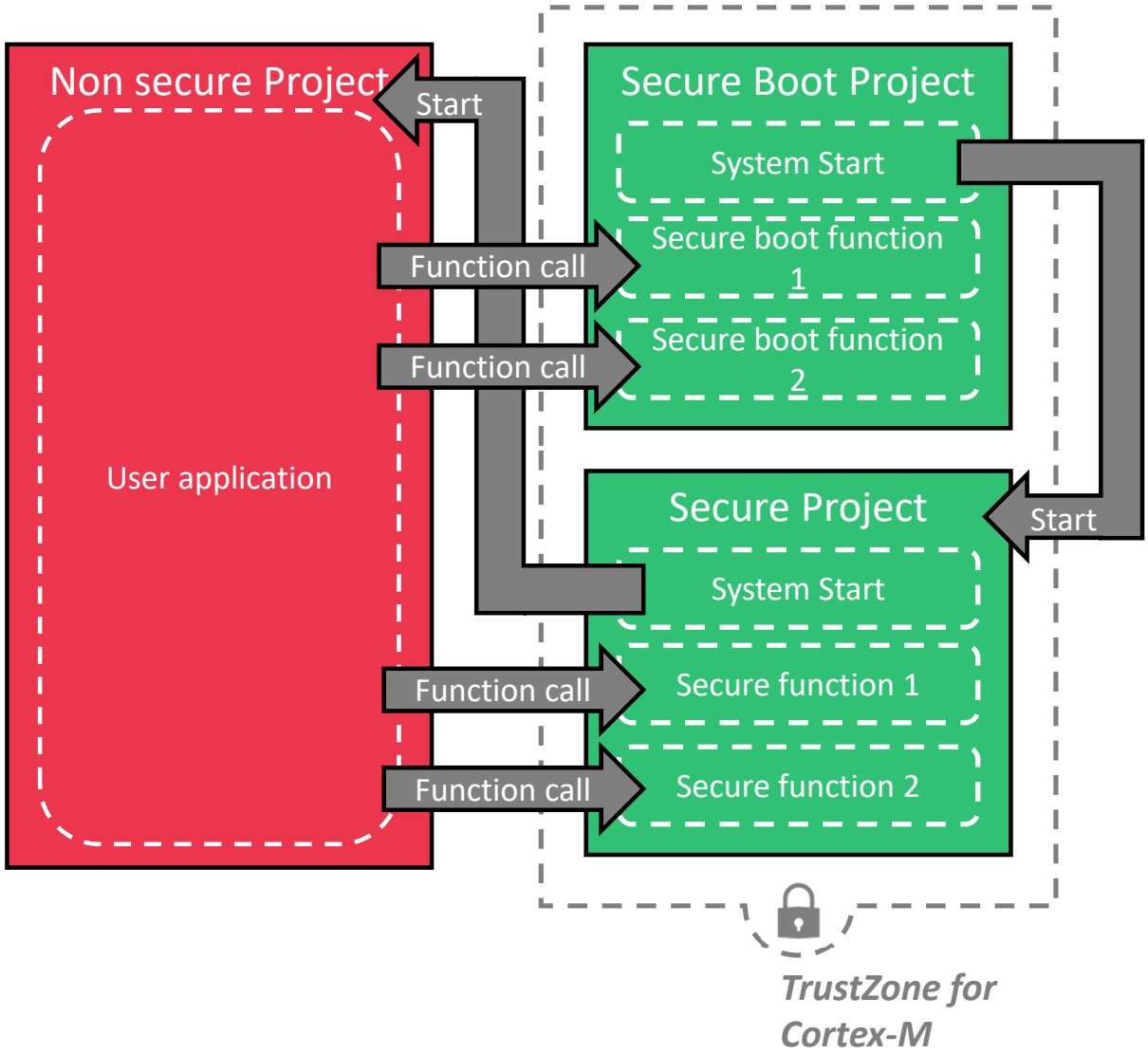


除了全片擦除保护之外，产品引导 ROM 还能够在执行前进行完整性检查或验证安全引导段中存储的固件。该验证机制是在部署和执行安全固件期间确保系统可信根而需考虑的关键要素。

2.5.1 创建具有引导程序的安全解决方案

为了方便开发具有安全引导程序的应用程序，Microchip Studio 7 提供了具有引导模板的预定义安全解决方案。该模板可用于评估和理解解决方案架构，以及着手开发具有安全引导项目的定制应用程序。下图给出了模板内容以及预配置项目之间的交互。

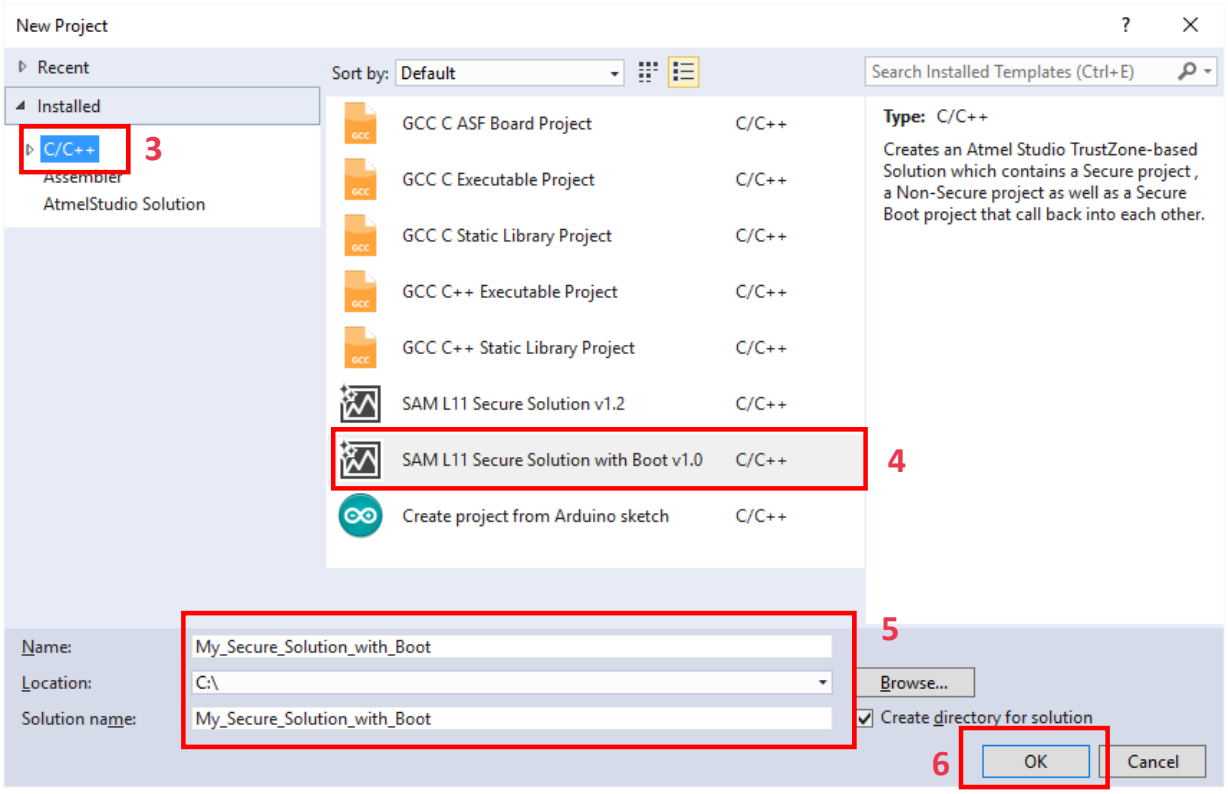
图 2-47. 安全解决方案模板内容



要使用 Microchip Studio 7 创建具有引导程序的安全解决方案，请按照以下步骤操作：

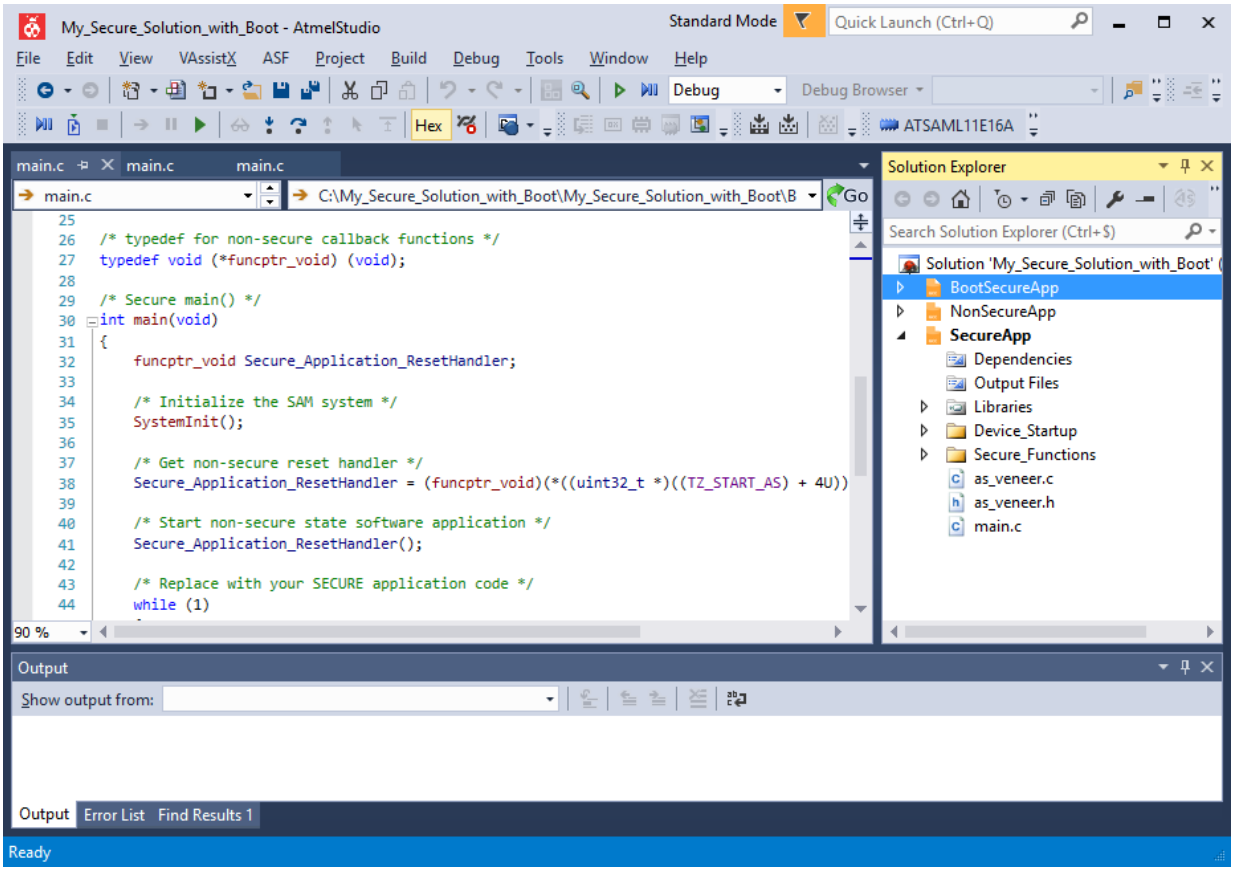
1. 打开 Microchip Studio 7。
2. 选择 *File* > *New* > *Project*。
3. 在 New Project 窗口中，执行以下操作来创建和配置一个新的安全解决方案：
   a. 展开 Installed，选择 C/C++。
   b. 选择 SAM L11 Secure Solution with Boot（具有引导的 SAM L11 安全解决方案）。
   c. 在 Name、Location 和 Solution name 中输入详细信息（相关示例见下图）。
   d. 单击 **OK**。

图 2-48. 创建具有引导的安全解决方案



创建后，解决方案将显示在 Microchip Studio 7 IDE 中，如下图所示：

图 2-49. 具有引导的安全解决方案



2.5.2 具有引导的安全解决方案模板说明

Microchip Studio 7 内提供的具有引导代码的 SAM L11 安全解决方案模板与前几章所述的 SAM L11 安全解决方案模板类似，只是嵌入了安全引导程序（存储在器件的 BS 存储区）。

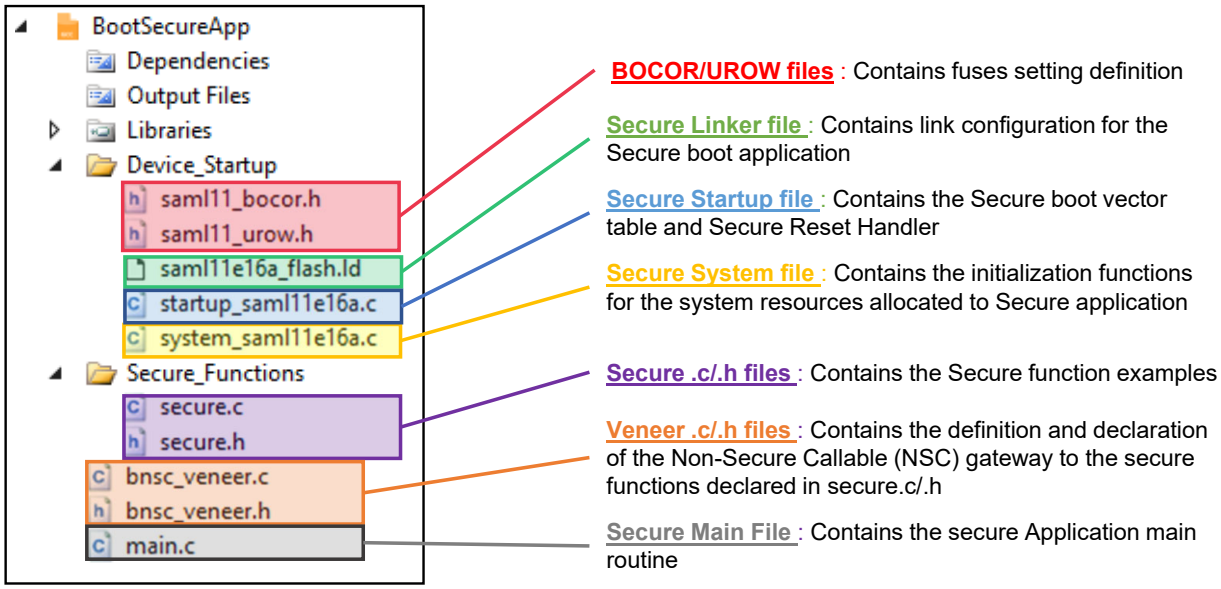
2.5.2.1 模板安全引导项目说明

解决方案模板中包含的安全引导项目旨在为基于 SAM L11 的安全引导代码开发提供预配置的开发库。该安全项目预配置为说明基于 SAM L11 的标准安全应用程序的以下方面：

- 安全引导函数示例的定义和声明
- 非安全区域的安全引导网关（模板）的定义和声明
- 对安全应用程序的安全调用

下图所示为预配置安全项目的文件架构：

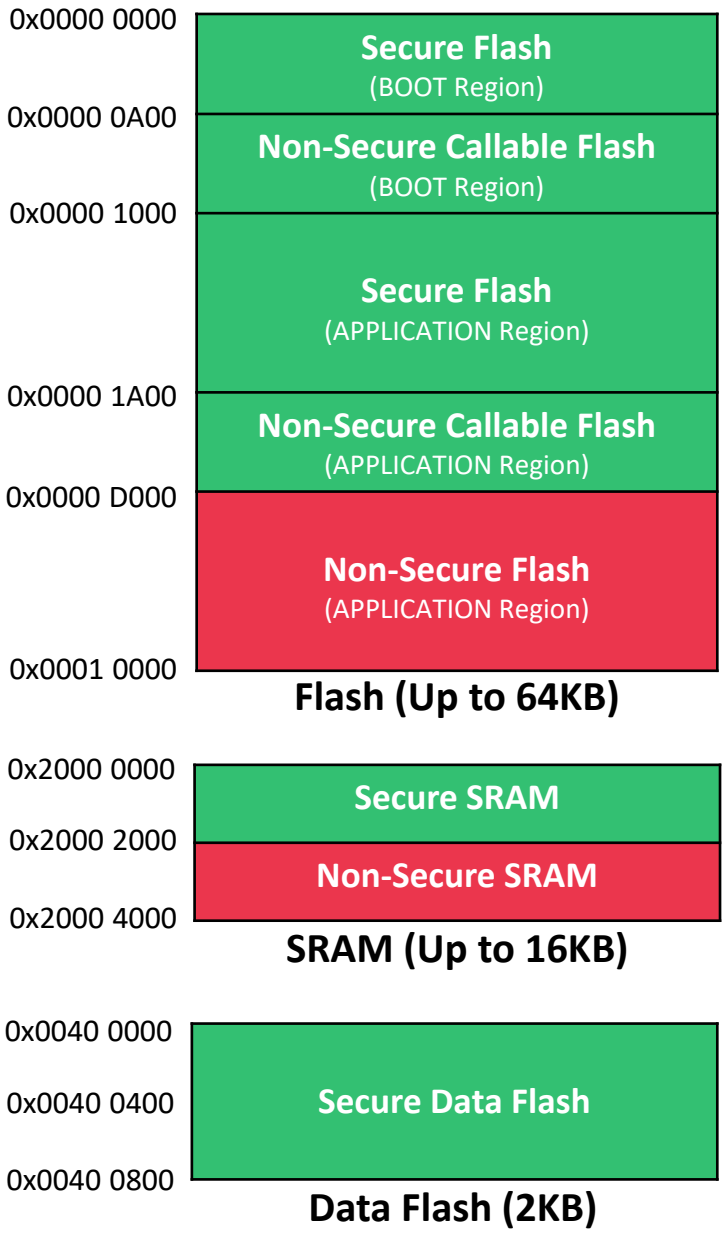
图 2-50. 安全引导项目架构



2.5.2.2 模板 NVM 熔丝配置

下图给出了默认的 USERROW 和 BOCOR 模板设置及相关的存储器映射。

图 2-51. 具有引导代码映射的默认安全解决方案



下表列出了 BOCOR 熔丝设置。

表 2-1. BOCOR 熔丝设置

熔丝	值	配置
BNSC	0x30	引导闪存非安全可调用大小 = BNSC*0x20 = 0x600
BS	0x10	引导闪存安全大小 = BS*0x100 = 0x1000
BOOTOPT	0x00	无安全引导验证
BOOTPROT	0x10	引导保护大小 = BOOTPROT*0x100 = 0x1000

.....（续）		
熔丝	**值**	**配置**
BCWEN	0x01	使能引导配置写操作
BCREN	0x01	使能引导配置读操作
CEKEY0	全 1	CE0 密钥 = 全 1
CEKEY1	全 1	CE1 密钥 = 全 1
CEKEY2	全 1	CE2 密钥 = 全 1
BOOTKEY	全 1	引导密钥 = 全 1

下表列出了 UROW 熔丝设置。

表 2-2. UROW 熔丝设置

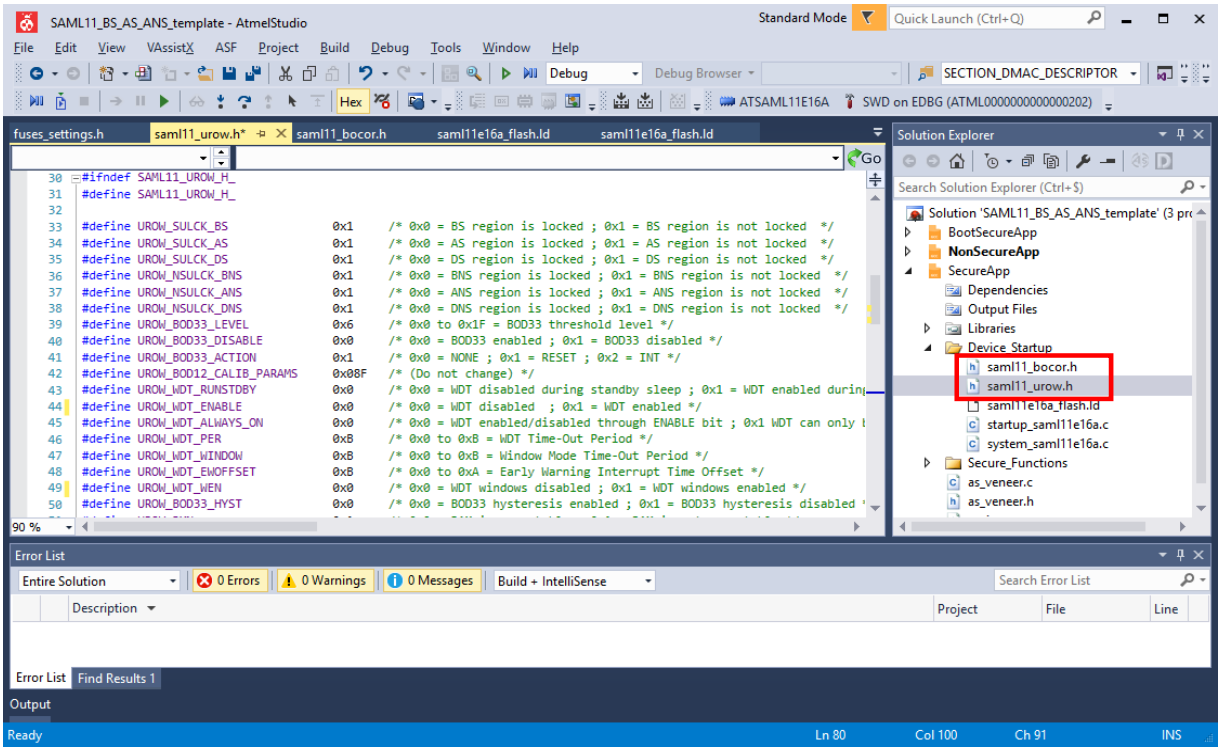
熔丝	值	配置
SULCK_BS	0x1	BS 区域未锁定
SULCK_AS	0x1	AS 区域未锁定
SULCK_DS	0x1	DS 区域未锁定
NSULCK_BNS	0x1	BNS 区域未锁定
NSULCK_ANS	0x1	ANS 区域未锁定
NSULCK_DNS	0x1	DNS 区域未锁定
BOD33_LEVEL	0x6	BOD33 阈值 = 0x6
BOD33_DISABLE	0x0	使能 BOD33
BOD33_ACTION	0x1	BOD 操作 = RESET
WDT_RUNSTDBY	0x0	在待机休眠期间禁止 WDT
WDT_ENABLE	0x0	禁止 WDT
WDT_ALWAYS_ON	0x0	通过 ENABLE 位使能/禁止 WDT
WDT_PER	0xB	WDT 超时周期 = 0xB
WDT_WINDOW	0xB	窗口模式超时周期 = 0xB
WDT_EWOFFSET	0xB	预警中断时间偏移 = 0xB
WDT_WEN	0x0	禁止 WDT 窗口
BOD33_HYST	0x0	无 BOD33 滞后
RXN	0x1	RAM 不可执行
DXN	0x1	数据闪存不可执行
AS	0x10	闪存应用程序安全大小 = AS*0x100 = 0x1000
ANSC	0x30	闪存应用程序非安全可调用大小 = ANSC*0x20 = 0x600
DS	0x08	数据闪存安全大小 = DS*0x100 = 0x800
RS	0x40	RAM 安全大小 = RS*0x80 = 0x2000
URWEN	0x1	使能用户行写操作

.....（续）

熔丝	值	配置
NONSECA	0x0000 0000	外设安全
NONSECB	0x0000 0000	外设安全
NONSECC	0x0000 0000	外设安全

为了方便定义和修改应用程序熔丝，所有熔丝值均在 `saml11_bocor.h` 和 `saml11_urow.h` 中定义，如下图所示。这些熔丝值可根据应用要求进行修改。

图 2-52. SAM L11 熔丝定义

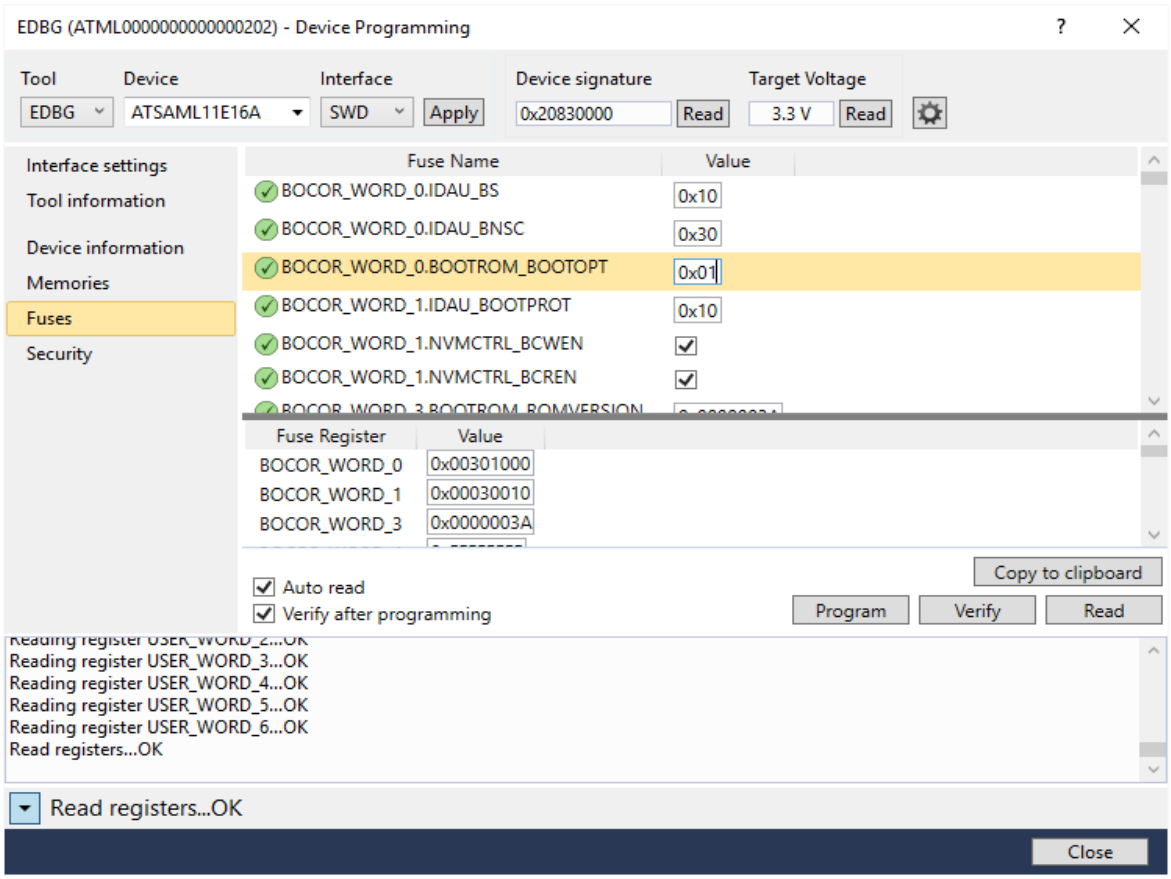


2.5.2.3 使能安全引导过程的 BS 验证

要使用 Microchip Studio 7 使能安全引导过程验证，请按照以下步骤操作：

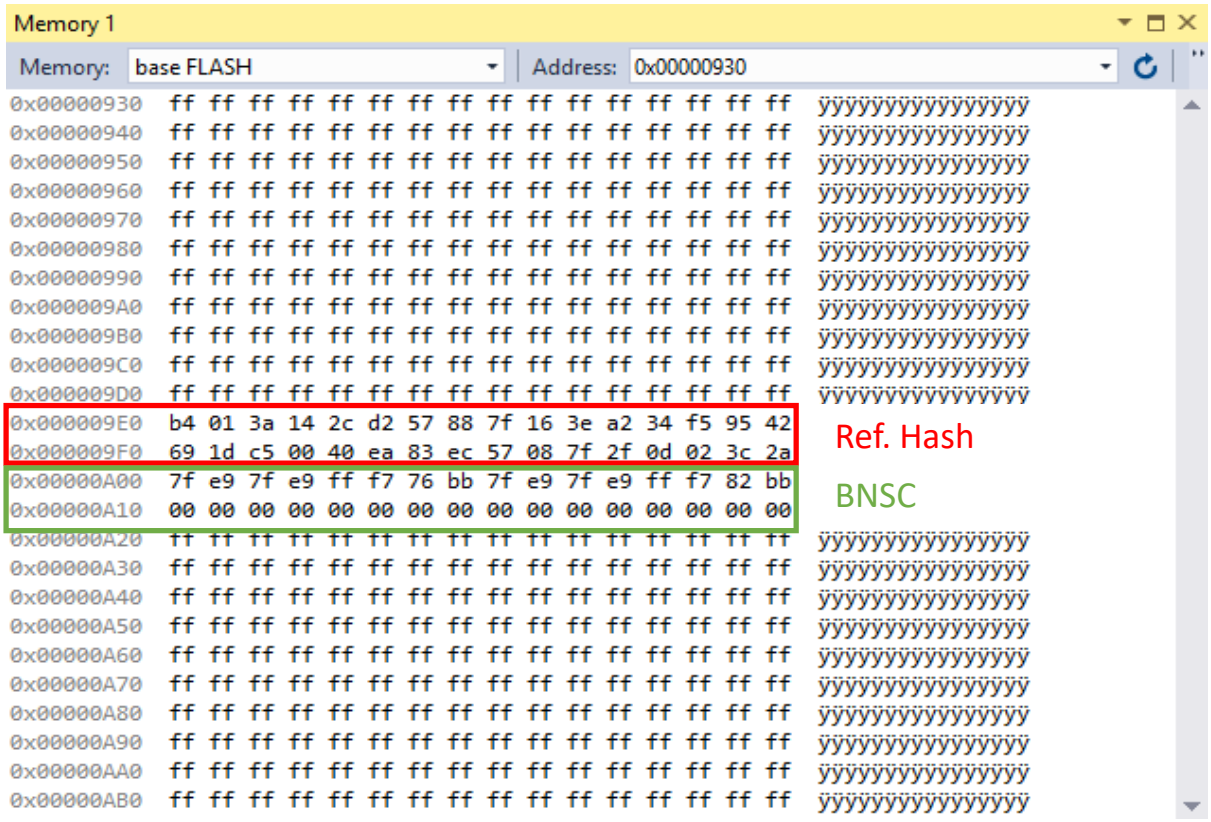
1. 使用器件编程执行 `ChipErase_ALL` 命令。
2. 使用 Microchip Studio IDE 编译引导应用程序。
3. 使用器件编程工具将 BOOTOPT 熔丝更改为 `0x01` 或 `0x02`。

图 2-53. 安全引导过程的 BS 验证



执行下图所示的步骤后，将计算参考哈希值并通过器件编程工具自动写入存储器。

图 2-54. 安全引导应用程序参考哈希值



3. 软件用例示例

3.1 非安全外设（TC0）

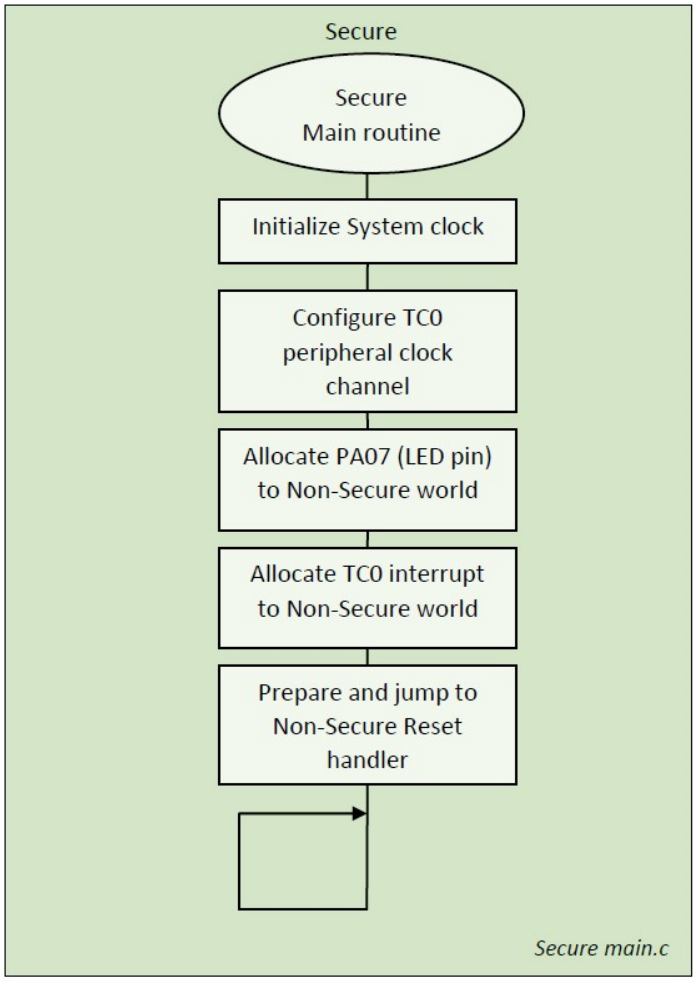
本用例示例介绍了如何将 SAM L11 集成外设（TC0）配置为非安全外设。

在本例中，安全项目负责将 PORT 和 TC 外设分配给非安全区域，设置系统时钟，然后跳转到非安全应用程序。

非安全应用程序使用 TC0 在 PA07 上生成 PWM 信号。

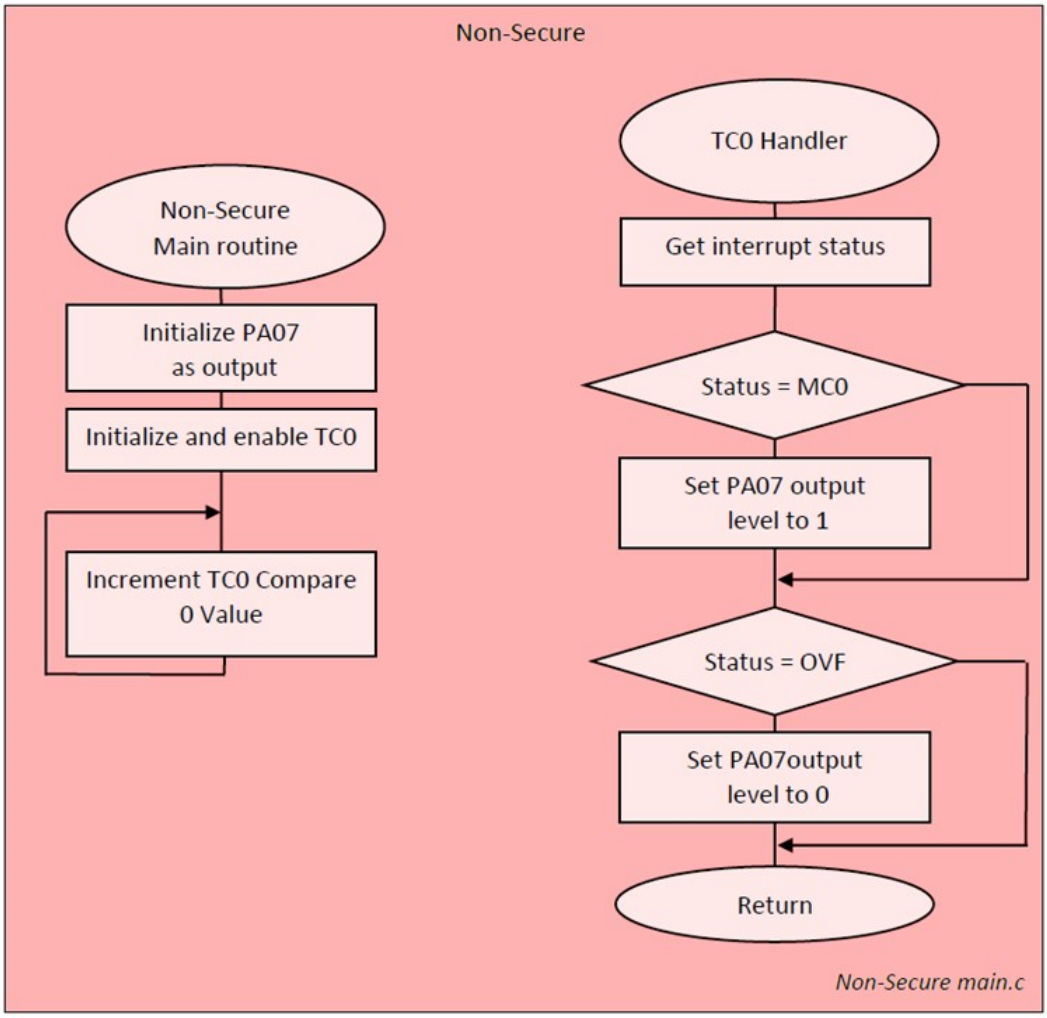
下图所示为安全主程序的执行流程。

图 3-1. 安全主程序流程图



下图所示为非安全主程序的执行流程。

图 3-2. 非安全主程序流程图



以下代码示例提供了用于将 TC0 及相关系统功能分配给非安全区域的关键安全区域函数调用和声明。

- 在熔丝定义（saml11_urow.h）中将 TC0 分配给非安全区域

```
...
#define UROW_NONSECC_SERCOM0     0x0     /* SERCOM0 安全 */
#define UROW_NONSECC_SERCOM1     0x0     /* SERCOM1 安全 */
#define UROW_NONSECC_SERCOM2     0x0     /* SERCOM2 安全 */
#define UROW_NONSECC_TC0         0x1     /* TC0 非安全 */
#define UROW_NONSECC_TC1         0x0     /* TC1 安全 */
#define UROW_NONSECC_TC2         0x0     /* TC2 安全 */
...
```

- 配置 TC0 外设时钟并将中断分配给非安全区域（安全应用程序）

```
int main(void)
{
uint32_t ret;
funcptr_void NonSecure_ResetHandler;

/* 初始化 SAM 系统 */
    SystemInit();
/* 配置 TC0 外设时钟通道 */
GCLK->PCHCTRL[14].reg =(GCLK_PCHCTRL_GEN(0) | GCLK_PCHCTRL_CHEN);
```

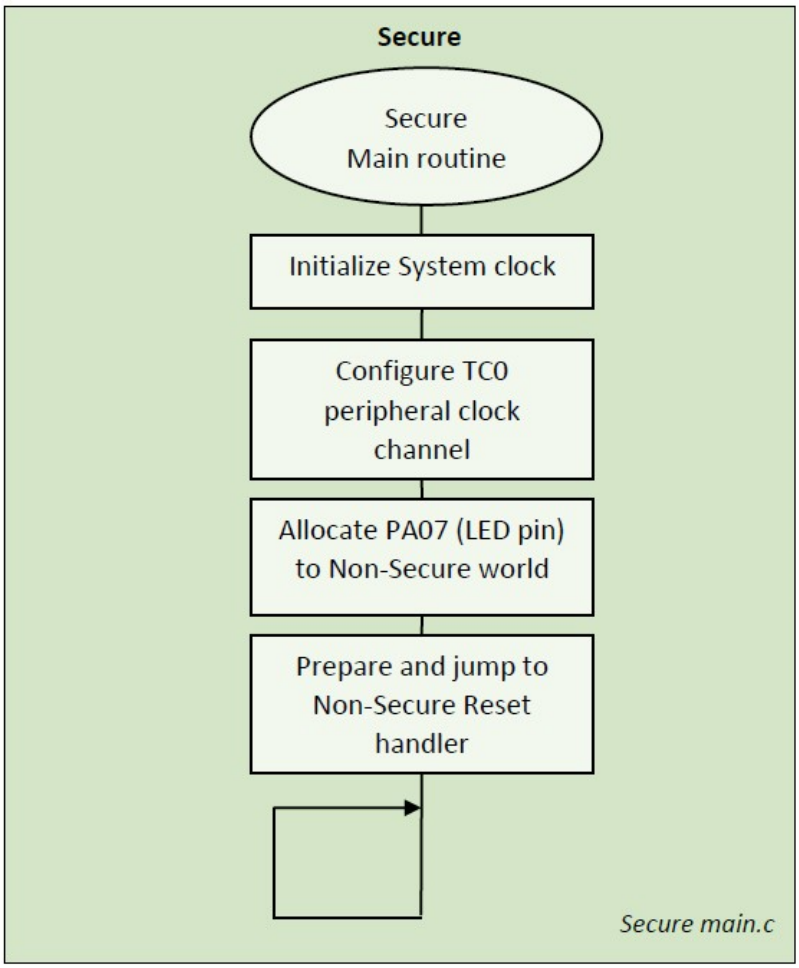
```
/* 将 PA07（LED 引脚）分配给非安全区域 */
    PORT_SEC->Group[0].NONSEC.reg = (PORT_PA07);
/* 将 TC0 中断分配给非安全区域 */
    NVIC_SetTargetState(TC0_IRQn);
/* 设置非安全主堆栈（MSP_NS） */
      TZ_set_MSP_NS(*((uint32_t *)(TZ_START_NS)));
/* 获取非安全复位处理程序 */
    NonSecure_ResetHandler = (funcptr_void)(*((uint32_t *)((TZ_START_NS) +
4U)));
/* 启动非安全状态软件应用程序 */
    NonSecure_ResetHandler();
    while (1)
    {
        NOP();
    }
}
```

3.2 安全外设（TC0）

本用例示例演示了如何将 SAM L11 集成外设（TC0）配置为安全外设。

在本用例中，安全项目负责配置系统资源和管理 TC 外设，以及将特定 TC0 API 和非安全回调提供给非安全区域。下图所示为安全主函数：

图 3-3. 安全主程序流程图



以下 API 或模板提供给非安全区域以驱动来自非安全区域的 TC0 外设：

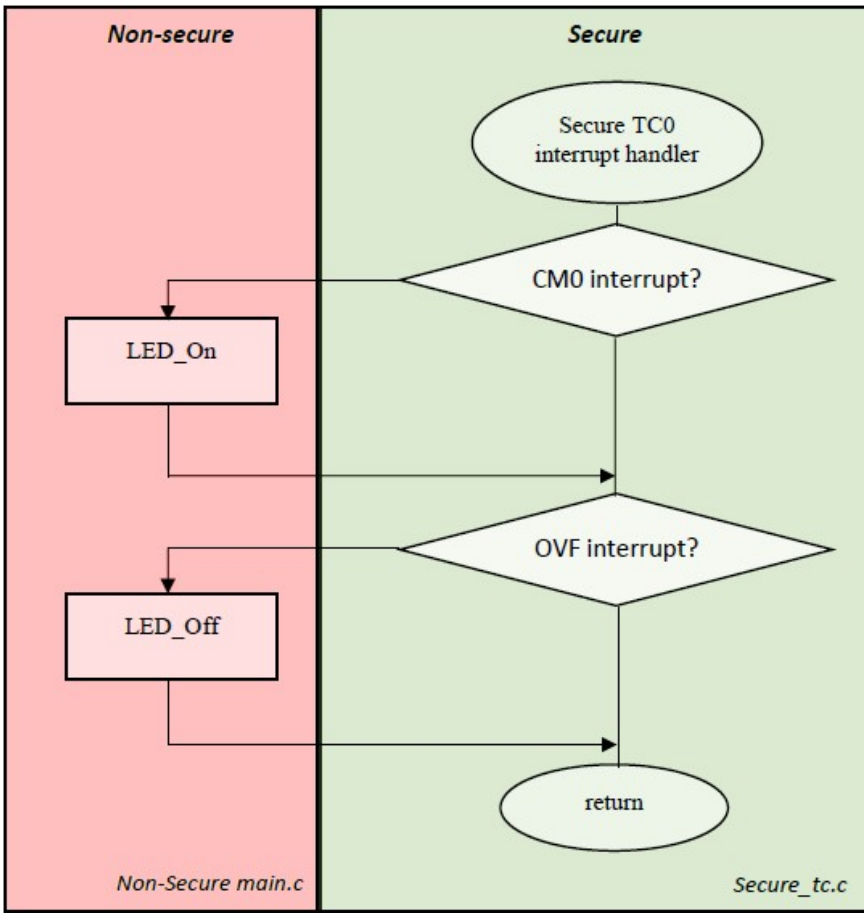
- tc0_compare_0_interrupt_callback_register(secure_void_cb_t pfunction);
- tc0_overflow_interrupt_callback_register(secure_void_cb_t pfunction);
- tc0_init(void);
- tc0_set_duty_cycle(uint8_t duty_cycle);

非安全区域通过安全区域提供的 API 和模板使用安全 TC0 并在 PA07 引脚上生成 PWM 信号。下图显示了应用程序的流程图以及与安全区域的交互。

图 3-4. 非安全主程序流程图

下图所示为安全 TC 处理程序。

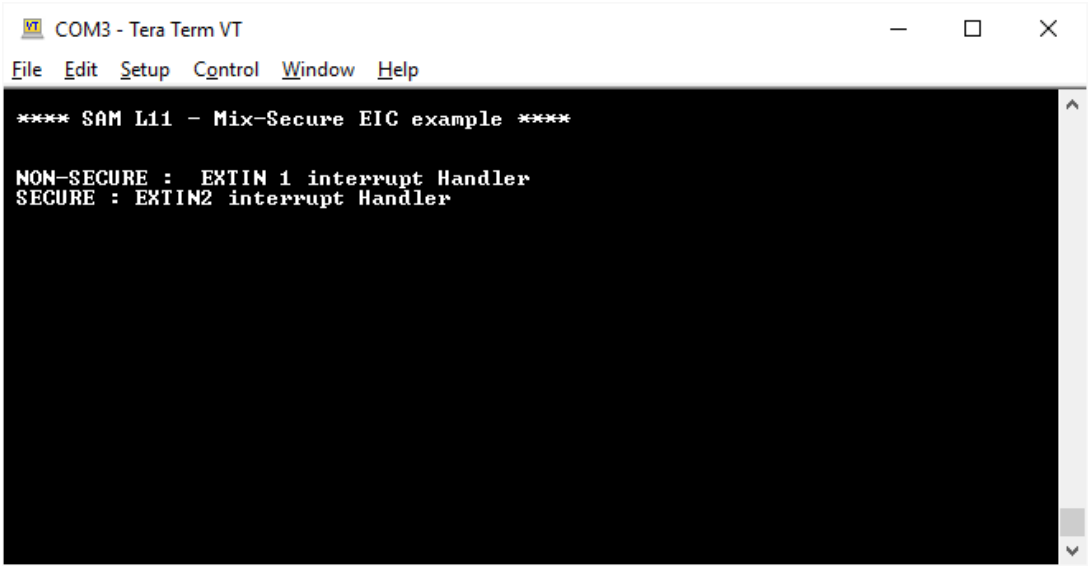
图 3-5. 安全 TC 处理程序流程图



3.3 混合安全外设（EIC）

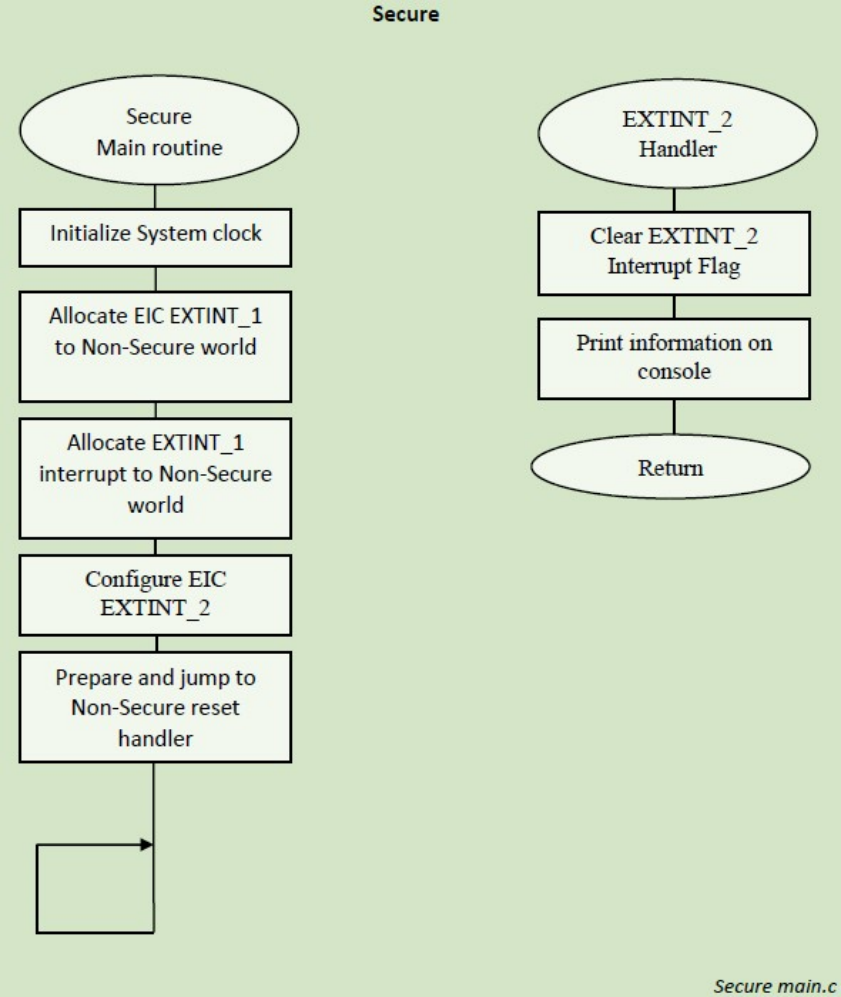
本用例示例介绍了如何配置和使用 SAM L11 混合安全外设（EIC）。使用本示例，用户可配置两条中断线（EXTIN 1 和 EXTIN2），然后将其分配给非安全区域和安全区域。这样可以在检测到 EXTIN 1 中断时执行非安全处理程序，在检测到 EXTIN 2 中断时执行安全处理程序，如下图所示。

图 3-6. 混合安全外设用例示例输出



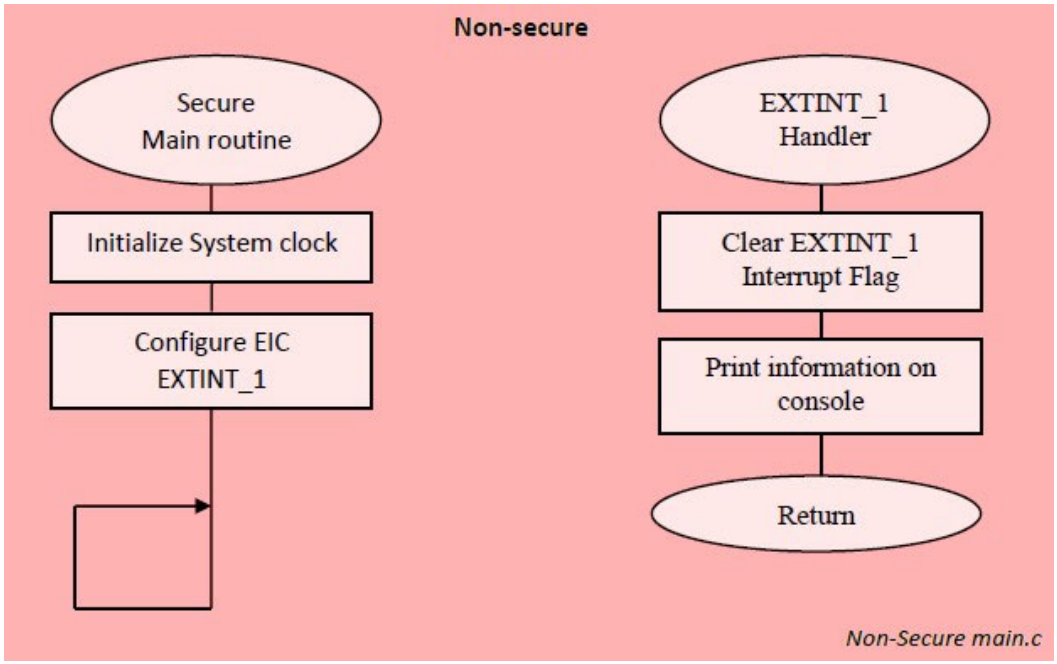
在本例中，安全项目负责配置系统资源，将 EIC 中断线 1 分配给非安全区域，以及管理安全中断线 2 上的外部中断。下图所示为安全主函数流程图。

图 3-7. 安全应用程序流程图



在本例中，非安全项目负责配置和处理 EIC 中断线 1，该中断线由安全应用程序分配给非安全区域。下图所示为相应的过程：

图 3-8. 非安全应用程序流程图



3.4 TrustRAM

SAM L11 中内置的 TrustRAM（TRAM）提供了以下高级安全功能来存储安全信息：

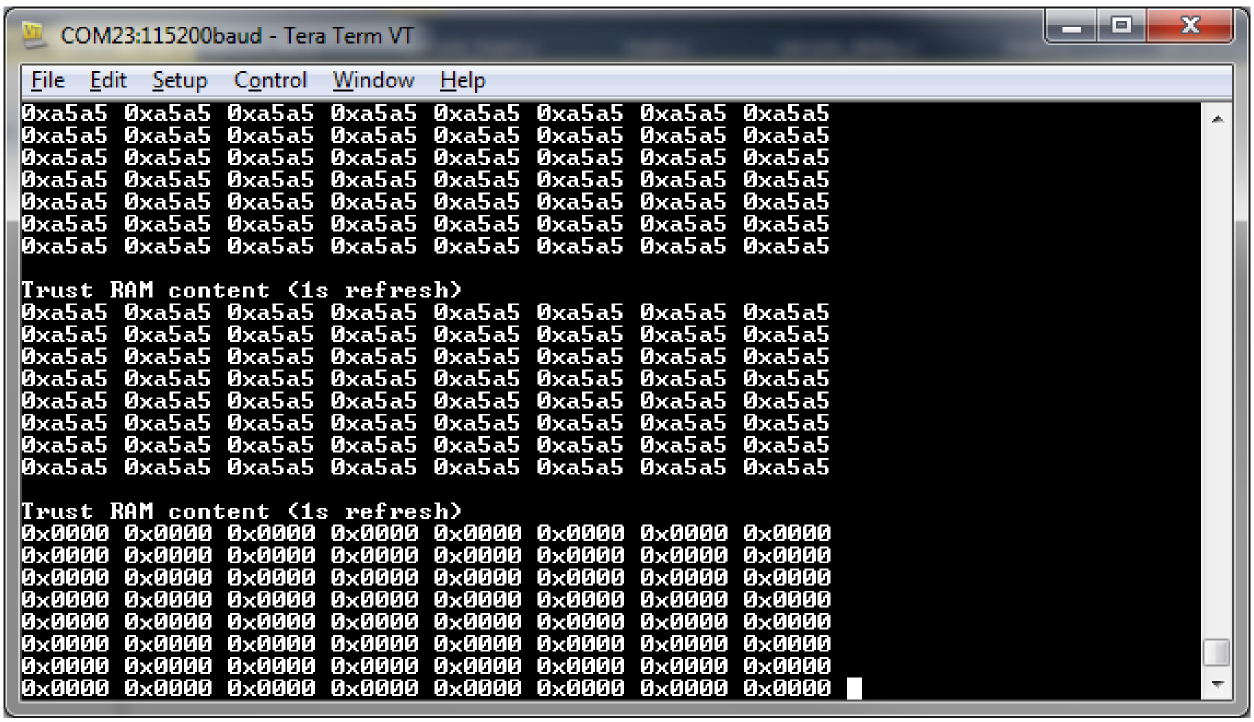
- 地址和数据加扰
- 静默访问
- 数据残留
- 主动屏蔽和篡改检测
- 在检测到篡改时将加扰密钥和 RAM 数据全部擦除

本文档随附的 TrustRAM 示例说明了如何配置 TrustRAM 的以下安全功能：

- 使用以下密钥激活地址和数据加扰： `0xCAFE`
- 使能静默访问
- 使能数据残留
- 在 PA8 上使能 RTC 静态篡改检测
- 使能在检测到篡改时将加扰密钥和 RAM 数据全部擦除

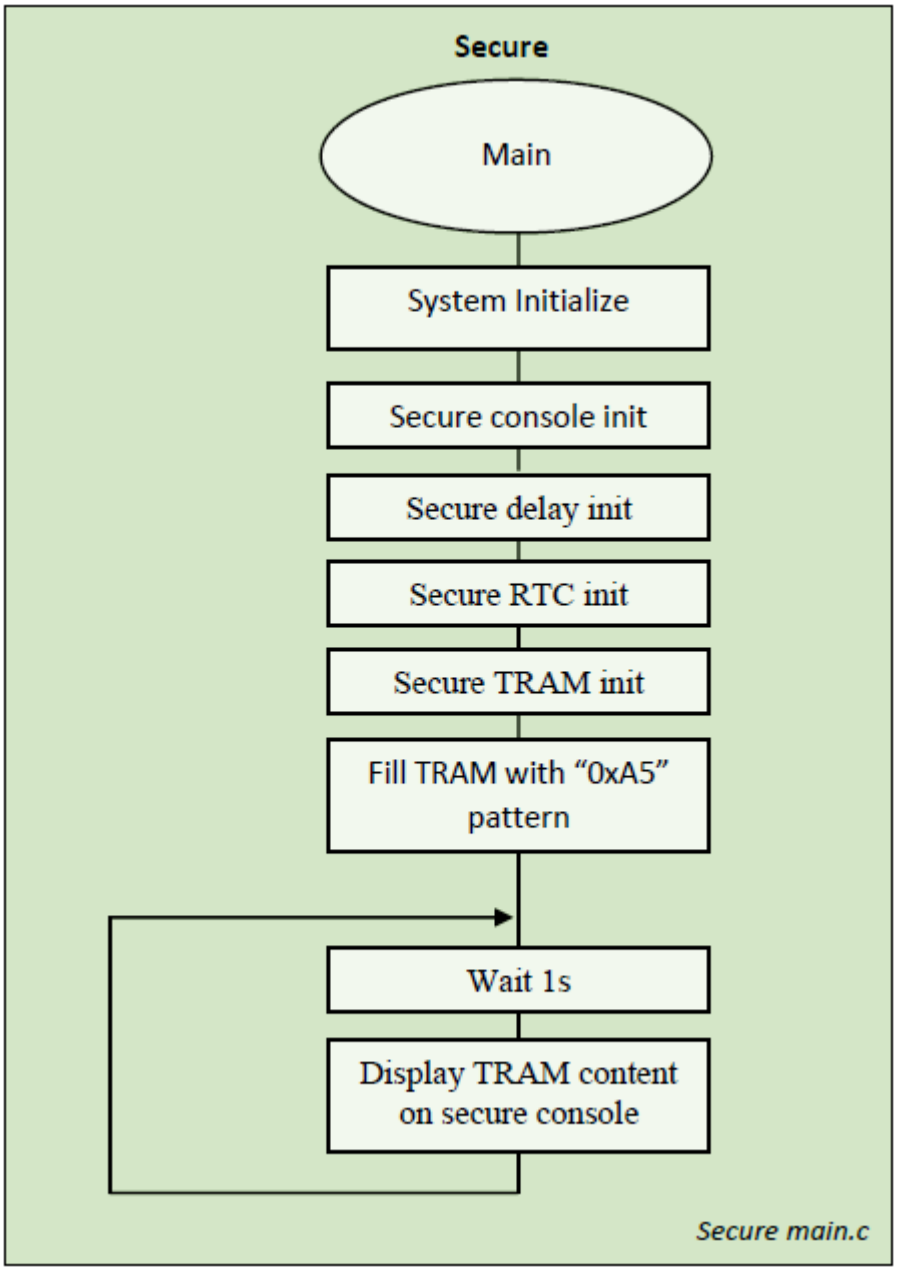
在本例中，TrustRAM 的内容显示在安全控制台（USART0）上，每秒刷新一次，允许用户进行静态和动态篡改检测以及 TrustRAM 全部擦除。

图 3-9. TRAM 用例应用程序输出



下图所示为 TRAM 的安全主函数。

图 3-10. TRAM 用例应用程序流程图



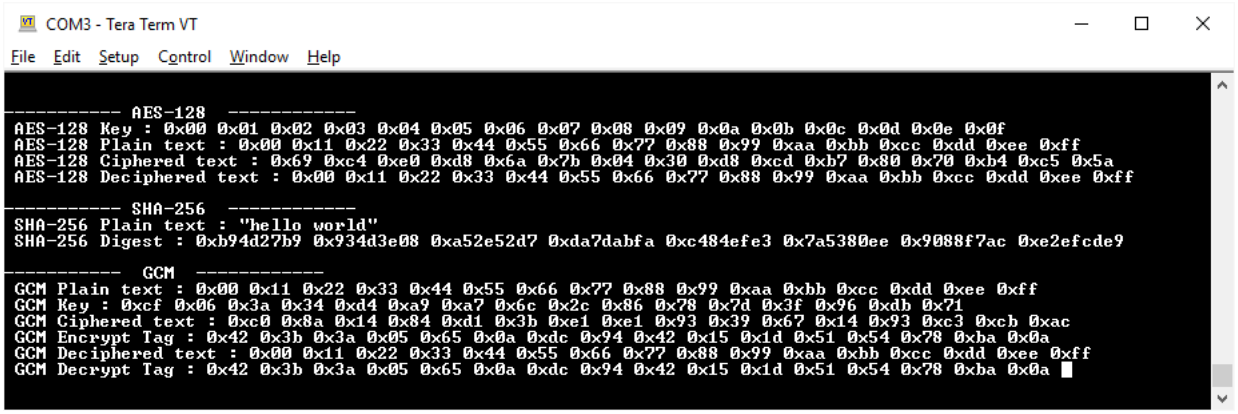
3.5 加密加速器（CRYA）

SAM L11 内置硬件加密加速器（CRYA）并在引导 ROM 中存储了相关软件函数，可为以下各项提供硬件加速：

- 高级加密标准（AES-128）加密和解密
- 安全哈希算法 2（SHA-256）身份验证
- Galois 计数器模式（Galois Counter Mode，GCM）加密和身份验证

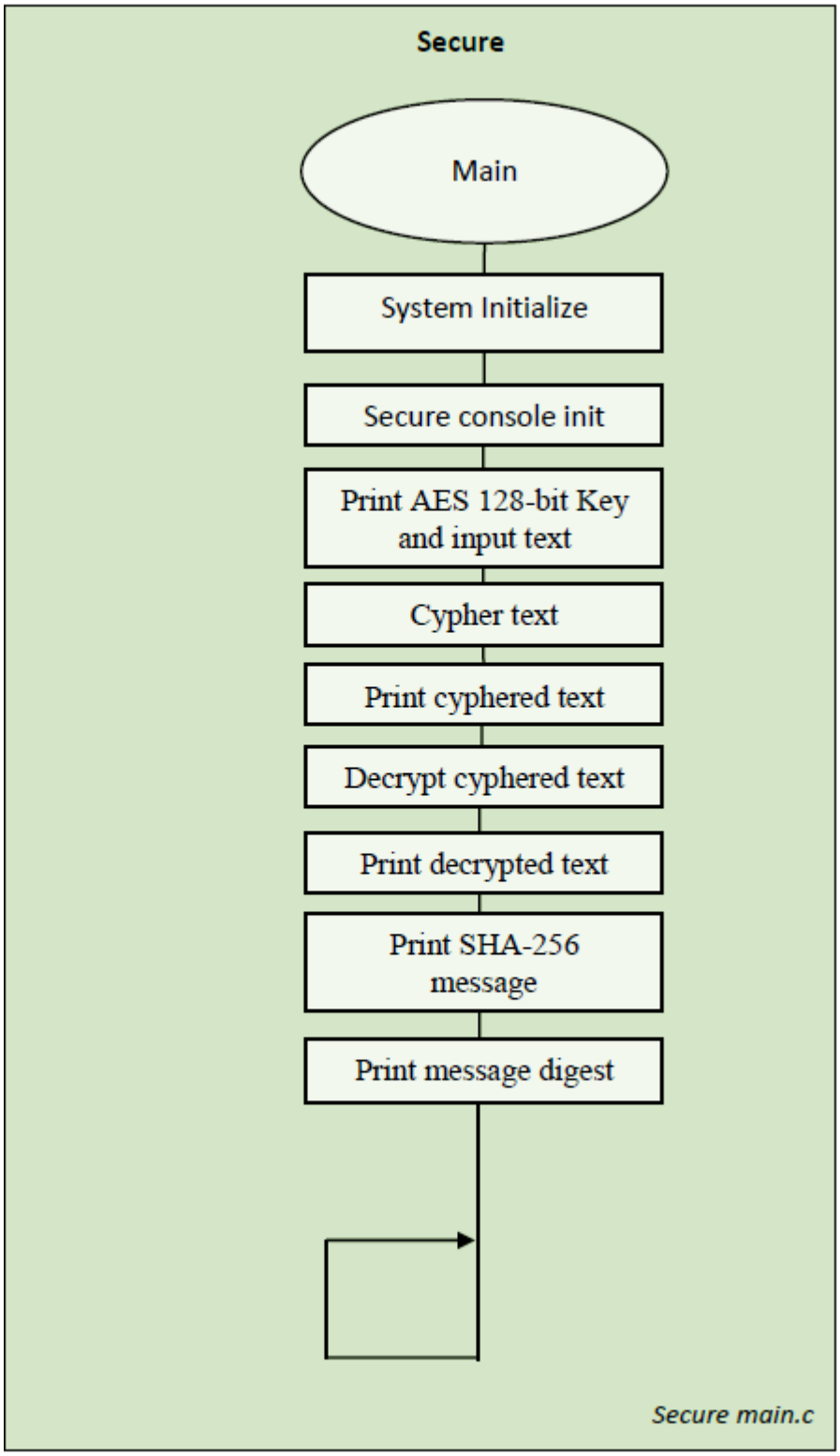
以下 CRYA 示例展示了如何将 CRYA 用于 AES 128 位密钥长度和 SHA-256 加密算法。

图 3-11. CRYA 用例应用程序输出



下图所示为 CRYA 用例应用程序流程图：

图 3-12. CRYA 用例应用程序流程图



3.6 数据闪存

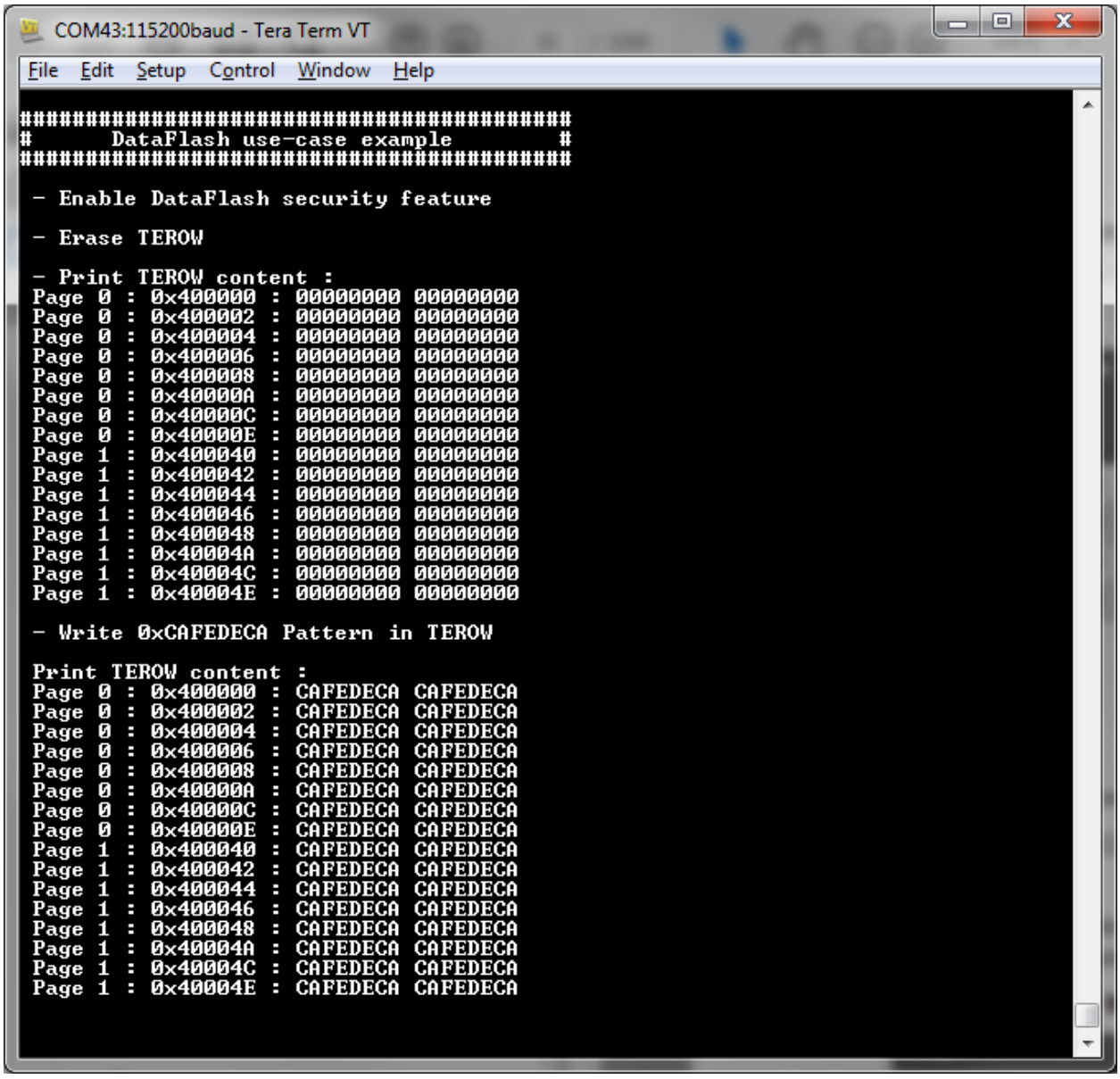
SAM L11 中内置的数据闪存提供了以下高级安全功能来存储安全信息：

- 数据加扰
- 选定行静默访问（TEROW）
- 在检测到篡改时擦除选定行（TEROW）

下图所示的数据闪存用例说明了如何配置 NVMCTRL 的安全数据闪存管理：

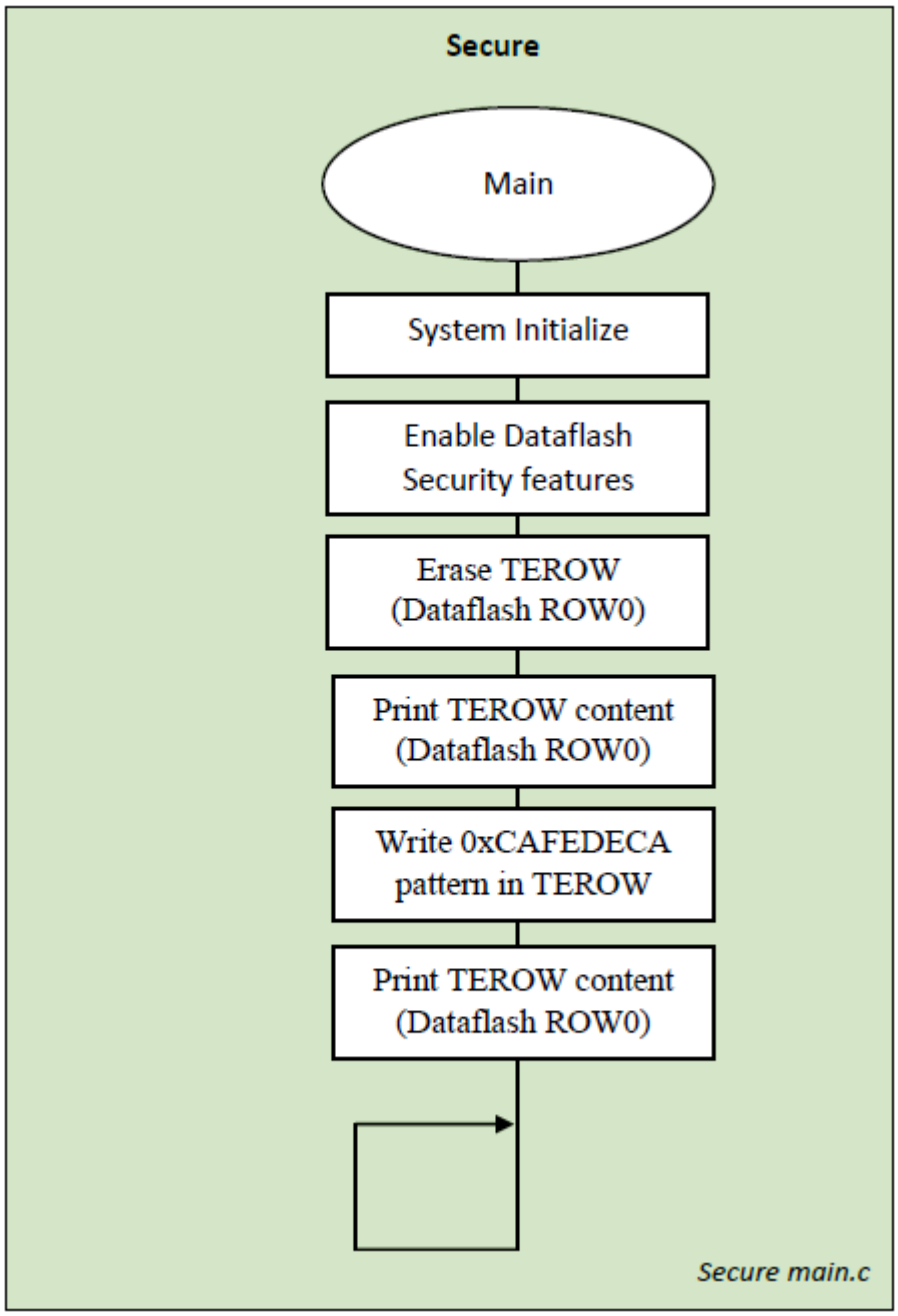
- 使用以下密钥激活数据加扰： `0x1234`
- 对第一个数据闪存行使能静默访问

图 3-13. 数据闪存用例应用程序输出



下图所示为数据闪存用例应用程序流程图：

图 3-14. 数据闪存用例应用程序流程图



4. 版本历史

版本 C——2022 年 7 月
本版本进行了以下更新：

- 将文档中的所有 Atmel 引用更新为 Microchip，并将所有 AS7 和 Atmel Studio 7 引用更新为 Microchip Studio 7
- 将文档中的所有*客户*引用更新为*开发人员*
- 更新了简介中 ARMv8 的标签
- 更新了 ARMv8-M 的 TrustZone 技术章节中的图 1-1、图 1-3 和 ARMv8 标签
- 更新了存储器安全属性章节中的图 1-4
- 更新了安全和非安全函数调用机制章节中的命名和术语
- 替换了非安全可调用 API 章节中的图 1-5
- 替换了非安全软件回调章节中的图 1-6 和图 1-7
- 替换了安全和非安全中断处理章节中的图 1-9 并为其重命名
- 调整了安全和非安全外设的结构并新增了图 1-10
- 增加了新的主题：外设安全属性
- 替换了混合安全外设（PAC 安全）章节中的图 1-11 和图 1-12
- 替换了混合安全外设（PAC 非安全）章节中的图 1-13
- 替换了调试访问级别（DAL）和全片擦除中图 1-14 和图 1-15 的图片
- 替换了安全引导章节中的图 1-20 并新增了表 SAML11 安全引导验证方法
- 为安全引导章节中的 BOOTKEY 新增了注释
- 替换了单开发人员方法章节中的图 2-1
- 替换了双开发人员方法章节中的图 2-2
- 替换了 NVM 行配置章节中的图 2-11
- 替换了调试安全解决方案章节中的图 2-16 并更新了标题
- 替换了调试安全解决方案章节中的图 2-17
- 替换了开发非安全项目（开发人员 B）章节中的图 2-22
- 替换了使项目链接器文件与 SAM L11 非安全存储器属性一致章节中的图 2-28
- 替换了将安全网关库添加并链接到非安全项目章节中的图 2-30
- 替换了添加并包含安全网关头文件章节中的图 2-39

版本 B——2019 年 4 月
调整了文档结构：

- 针对开发安全应用程序新增以下章节：开发具有安全引导程序的解决方案（客户 A）
- 删除了主题*部署具有安全和非安全自举程序的应用程序*，并将其内容纳入 SAM L11 安全功能简介
- 删除了主题*如何定义和使用安全和非安全外设*，并将其内容纳入软件用例示例
- 重新编写了简介，以将更新内容反映到文档中。

更新了以下章节：

- 更新了 ARMv8-M 架构的 TrustZone 技术章节，新增了图片
- 更新了安全和非安全外设章节，新增了图片
- 更新了混合安全集成外设章节，新增了图片
- 更新了调试访问级别（DAL）和全片擦除章节，新增了图片
- 更新了安全引导章节，新增了图片
- 更新了单开发人员方法章节，新增了图片
- 更新了双开发人员方法章节，新增了图片
- 更新了开发安全解决方案（客户 A）章节，新增了图片

- 更新了开发非安全项目（客户 B）章节，新增了图片
- 更新了非安全外设章节，新增了图片和代码块
- 更新了安全外设章节，新增了图片
- 更新了混合安全外设章节，新增了图片
- 更新了 TrustRAM（TRAM）章节，新增了图片
- 更新了加密加速器（CRYA）章节，新增了图片
- 更新了数据闪存章节，新增了图片

版本 A——2018 年 6 月
本文档的初始版本。

Microchip 网站

Microchip 网站（www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。我们的网站提供以下内容：

- **产品支持**——数据手册和勘误表、应用笔记和示例程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及归档软件
- **一般技术支持**——常见问题解答（FAQ）、技术支持请求、在线讨论组以及 Microchip 设计伙伴计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

产品变更通知服务

Microchip 的产品变更通知服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请访问 www.microchip.com/pcn，然后按照注册说明进行操作。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师（ESE）
- 技术支持

客户应联系其代理商、代表或 ESE 寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过 www.microchip.com/support 获得网上技术支持。

Microchip 器件代码保护功能

请注意以下有关 Microchip 产品代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术规范。
- Microchip 确信：在正常使用且符合工作规范的情况下，Microchip 系列产品非常安全。
- Microchip 注重并积极保护其知识产权。严禁任何试图破坏 Microchip 产品代码保护功能的行为，这种行为可能会违反《数字千年版权法案》（Digital Millennium Copyright Act）。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是“牢不可破”的。代码保护功能处于持续发展中。Microchip 承诺将不断改进产品的代码保护功能。

法律声明

提供本文档的中文版本仅为了便于理解。请勿忽视文档中包含的英文部分，因为其中提供了有关 Microchip 产品性能和使用情况的有用信息。Microchip Technology Inc.及其分公司和相关公司、各级主管与员工及事务代理机构对译文中可能存在的任何差错不承担任何责任。建议参考 Microchip Technology Inc.的英文原版文档。

本出版物及其提供的信息仅适用于 Microchip 产品，包括设计、测试以及将 Microchip 产品集成到您的应用中。以其他任何方式使用这些信息都将被视为违反条款。本出版物中的器件应用信息仅为您提供便利，将来可能会发生更新。如需额外的支持，请联系当地的 Microchip 销售办事处，或访问 https://www.microchip.com/en-us/support/design-help/client-supportservices。

商标

ISBN： 978-1-6683-1043-4

质量管理体系

有关 Microchip 的质量管理体系的信息，请访问 www.microchip.com/quality。

全球销售及服务网点

美洲	亚太地区	亚太地区	欧洲
公司总部 2355 West Chandler Blvd. Chandler, AZ 85224-6199 电话：480-792-7200 传真：480-792-7277 技术支持： www.microchip.com/support 网址： www.microchip.com **亚特兰大** 德卢斯，佐治亚州 电话：678-957-9614 传真：678-957-1455 **奥斯汀，德克萨斯州** 电话：512-257-3370 **波士顿** 韦斯特伯鲁，马萨诸塞州 电话：774-760-0087 传真：774-760-0088 **芝加哥** 艾塔斯卡，伊利诺伊州 电话：630-285-0071 传真：630-285-0075 **达拉斯** 阿迪森，德克萨斯州 电话：972-818-7423 传真：972-818-2924 **底特律** 诺维，密歇根州 电话：248-848-4000 **休斯顿，德克萨斯州** 电话：281-894-5983 **印第安纳波利斯** 诺布尔斯维尔，印第安纳州 电话：317-773-8323 传真：317-773-5453 电话：317-536-2380 **洛杉矶** 米慎维荷，加利福尼亚州 电话：949-462-9523 传真：949-462-9608 电话：951-273-7800 **罗利，北卡罗来纳州** 电话：919-844-7510 **纽约，纽约州** 电话：631-435-6000 **圣何塞，加利福尼亚州** 电话：408-735-9110 电话：408-436-4270 **加拿大 - 多伦多** 电话：905-695-1980 传真：905-695-2078	**澳大利亚 - 悉尼** 电话：61-2-9868-6733 **中国 - 北京** 电话：86-10-8569-7000 **中国 - 成都** 电话：86-28-8665-5511 **中国 - 重庆** 电话：86-23-8980-9588 **中国 - 东莞** 电话：86-769-8702-9880 **中国 - 广州** 电话：86-20-8755-8029 **中国 - 杭州** 电话：86-571-8792-8115 **中国 - 香港特别行政区** 电话：852-2943-5100 **中国 - 南京** 电话：86-25-8473-2460 **中国 - 青岛** 电话：86-532-8502-7355 **中国 - 上海** 电话：86-21-3326-8000 **中国 - 沈阳** 电话：86-24-2334-2829 **中国 - 深圳** 电话：86-755-8864-2200 **中国 - 苏州** 电话：86-186-6233-1526 **中国 - 武汉** 电话：86-27-5980-5300 **中国 - 西安** 电话：86-29-8833-7252 **中国 - 厦门** 电话：86-592-2388138 **中国 - 珠海** 电话：86-756-3210040	**印度 - 班加罗尔** 电话：91-80-3090-4444 **印度 - 新德里** 电话：91-11-4160-8631 **印度 - 浦那** 电话：91-20-4121-0141 **日本 - 大阪** 电话：81-6-6152-7160 **日本 - 东京** 电话：81-3-6880-3770 **韩国 - 大邱** 电话：82-53-744-4301 **韩国 - 首尔** 电话：82-2-554-7200 **马来西亚 - 吉隆坡** 电话：60-3-7651-7906 **马来西亚 - 槟榔屿** 电话：60-4-227-8870 **菲律宾 - 马尼拉** 电话：63-2-634-9065 **新加坡** 电话：65-6334-8870 **台湾地区 - 新竹** 电话：886-3-577-8366 **台湾地区 - 高雄** 电话：886-7-213-7830 **台湾地区 - 台北** 电话：886-2-2508-8600 **泰国 - 曼谷** 电话：66-2-694-1351 **越南 - 胡志明市** 电话：84-28-5448-2100	**奥地利 - 韦尔斯** 电话：43-7242-2244-39 传真：43-7242-2244-393 **丹麦 - 哥本哈根** 电话：45-4485-5910 传真：45-4485-2829 **芬兰 - 埃斯波** 电话：358-9-4520-820 **法国 - 巴黎** 电话：33-1-69-53-63-20 传真：33-1-69-30-90-79 **德国 - 加兴** 电话：49-8931-9700 **德国 - 哈恩** 电话：49-2129-3766400 **德国 - 海尔布隆** 电话：49-7131-72400 **德国 - 卡尔斯鲁厄** 电话：49-721-625370 **德国 - 慕尼黑** 电话：49-89-627-144-0 传真：49-89-627-144-44 **德国 - 罗森海姆** 电话：49-8031-354-560 **以色列 - 若那那市** 电话：972-9-744-7705 **意大利 - 米兰** 电话：39-0331-742611 传真：39-0331-466781 **意大利 - 帕多瓦** 电话：39-049-7625286 **荷兰 - 德卢内市** 电话：31-416-690399 传真：31-416-690340 **挪威 - 特隆赫姆** 电话：47-72884388 **波兰 - 华沙** 电话：48-22-3325737 **罗马尼亚 - 布加勒斯特** 电话：40-21-407-87-50 **西班牙 - 马德里** 电话：34-91-708-08-90 传真：34-91-708-08-91 **瑞典 - 哥德堡** 电话：46-31-704-60-40 **瑞典 - 斯德哥尔摩** 电话：46-8-5090-4654 **英国 - 沃金厄姆** 电话：44-118-921-5800 传真：44-118-921-5820